RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Université de Toulouse

Doctorat de l'Université de Toulouse

préparé à l'Université Toulouse - Jean Jaurès

Routage et localisation en intérieur dans les réseaux IoT : approches intelligentes basées sur les métaheuristiques et l'apprentissage profond

---

Thèse présentée et soutenue, le 10 juillet 2024 par

Sihem TLILI

École doctorale
EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse

Spécialité
Informatique et Télécommunications

Unité de recherche
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Thierry VAL

Composition du jury
M. Adrien VAN DEN BOSSCHE, Président, Université Toulouse - Jean Jaurès
M. Gérard CHALHOUB, Rapporteur, Université Clermont Auvergne
Mme Hanen IDOUDI, Rapporteure, École nationale des sciences de l'informatique
Mme Cassandre VEY, Examinatrice, Université Toulouse - Jean Jaurès
M. Thierry VAL, Directeur de thèse, Université Toulouse - Jean Jaurès
M. Sami MNASRI, Co-encadrant de thèse, Université de Tabuk Arabie Saoudite

Routing and Indoor Localization in IoT Networks: Intelligent Approaches Based on Metaheuristics and Deep Learning

Acknowledgements

I would first like to express my deep respect and gratitude to **Pr. Thierry VAL**, my thesis director, for his support, availability, and understanding. In particular, I am grateful to him for giving me significant freedom in my research, and carefully considering all my comments and ideas and for answering my numerous questions. His guidance and mentorship were very helpful to me in completing this work. I have really appreciated his support and enjoyed our conversations.

I sincerely thank my co-supervisor, **Dr. Sami MNASRI**, for his interest in this work. I am deeply grateful to him for his invaluable guidance and encouragement. I have really appreciated his support throughout the thesis process.

I also thank **Pr. Gérard CHALHOUB** and **Dr. Hanen IDOUDI** for agreeing to be part of my evaluation committee.

I would also like to thank **Pr. Adrien VAN DEN BOSSCHE** and **Dr. Cassandre VEY** for agreeing to examine my thesis work.

I also thank **Dr. Cassandre VEY** and **Dr. Guillaume GAILLARD** for their valuable assistance.

Contents

List of Figures

List of Algorithms

List of Tables

List of acronyms

AcNSGA-III Ant-Colony Non-dominated Sorting Genetic Algorithm III

ACO Ant Colony Optimization

AENN Auto-encoder Neural Network

ANN Artificial Neural Network

AOA Arithmetic Optimizer Algorithm

AoA Angle of arrival

API Application Programming Interface

APIT Adaptive Point-In-Triangulation

BBPSO Bare Bones Particle Swarm Optimization

BFO Bacteria Foraging Optimization

BFOA Bacteria Foraging Optimization Algorithm

BGD Batch Gradient Descent

BLE Bluetooth Low Energy

CNN Convolutional Neural Network

CPE Convex Position Estimation

CSI Channel State Information

D Diversity

DL Deep Learning

DL-GWO-ToF Deep Learning-Gray Wolf Optimizer-Time of Flight localization solution

DLH Dimension learning-based hunting

DMOPSO Dual Multi-Objective Particle Swarm Optimization

DNN Deep Neural Network

DV-HoP Distance Vector-Hop

EMOPSOC Enhanced Multi-Objective Particle Swarm Optimization with Clustering

EMSA Extended Multi-objective Simplex Algorithm

EP Evolutionary programming

ER Estimated Rectangle

ESPEA Electrostatic Potential Energy Evolutionary Algorithm

FND First Node Dies

FOA Firefly Optimization Algorithm

FSSP Flow Shop Scheduling Problem

FTR Follow the Regularized Leader

GD Gradient Descent

GNSS Global Navigation Satellite Systems

GRU Gated Recurrent Unit network

GWO Gray Wolf Optimizer

HV Hypervolume

I-GWO Improved Gray Wolf Optimizer

IGD Inverted Generational Distance

IMOGWO Improved Multi-objective Gray Wolf Optimizer

IoT Internet of Things

IRIT Institut de Recherche en Informatique de Toulouse

IWCL Improved Weighted Centroid Localization Algorithm

IWDOA Intelligent Water Drop Optimization Algorithm

LMOP Linear Multi-objective Optimization Problem

LP Linear Programming

LSTM Long Short-Term Memory

M2M Machine to Machine

MAE Mean Absolute Error

MARE Mean Absolute Relative Error

MCLP Maximal Covering Location Problem

MFO Moth-Flame Optimization

MIQP Mixed-Integer Quadratic Programming

ML Machine Learning

MLP Multi-layer Perceptron

MoCA Multimedia over Coax Alliance

MOEA/D Multi-objective Evolutionary Algorithm Based on Decomposition

MOEA/D-Eps Epsilon Multi-objective Evolutionary Algorithm Based on Decomposition

MOEA/D-IEps Improved Epsilon Multi-objective Evolutionary Algorithm Based on Decomposition

MOEA/DD Multi-Objective Evolutionary Algorithm based on Decomposition and Dominance

MOGWO Multi-objective Gray Wolf Optimizer

MOP Multi-objective Optimization Problems

MOPSO Multi-objective Particle Swarm Optimization

MQTT Message Queuing Telemetry Transport

MS Maximum Spread

MSE Mean Squared Error

NHV Normalized Hypervolume

NSGA-III Non-dominated Sorting Genetic Algorithm III

OTSA Orikaeshi Tanren Simulated Annealing

PDR Packet Delivery Ratio

PIT Point-In-Triangulation

PSCLP Partial Set Covering Location Problem

PSO Particle Swarm Optimization

QoS Quality of Service

QPSO Quantum-behaved Particle Swarm Optimization

R2 R-squared

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

RSSI Received Signal Strength Indicator

SA Simulated Annealing

SDE Standard-Deviation of Energy

SFO Sun Flower Optimization

SGD Stochastic Gradient Descent

SNR Signal-to-Noise Ratio

SP Spacing Metric

TDoA Time Difference of Arrival

ToF Time of flight

TS Tabu Search

TWR Two-Way Ranging

UWB Ultra-Wide Band

WFG Walking Fish Group

WFGHV Walking Fish Group Hypervolume

Chapter 1

Introduction and overview

Contents

1.1 Motivations and problematic

In recent years, the Internet of Things (IoT) has expanded intensively and has been utilized in several fields. Its wide use and various applications have revolutionized the way we interact with the world around us. By offering diverse possibilities, IoT has also transformed many aspects of our daily life.

However, this considerable expansion has lead to the emergence of several challenges. For instance, we can cite the routing problem in IoT networks having mesh structure, especially due to their dynamic nature and constant topological scalability. In fact, routing consists in determining the optimal paths followed to transfer data between the different devices connected within the network. Efficient routing in IoT networks ensure smooth, reliable and energy-efficient communications, which guarantee the smooth functioning of IoT applications. First, it allows optimizing the performance of the IoT applications by reducing transmission delays and maximizing the use of the network resource. Additionally, efficient routing increases communication reliability by choosing reliable paths, which minimizes the risks of data loss or service interruptions between IoT devices. It also ensures significant energy saving, particularly in battery-powered devices, by limiting unnecessary transmissions or favoring energy-efficient paths. Moreover, efficient routing allows the IoT network to dynamically adapt to topology changes and evolve to integrate new devices, which

makes it more flexible and scalable. Besides, it protects data by applying secure routing strategies to prevent attacks and protect sensitive information sent through the IoT network.
Another important challenge in IoT networks is that of indoor localization which consists in determining the position of the connected devices inside an enclosed indoor space. Unlike outdoor localization systems, traditional Global Navigation Satellite Systems (GNSS)-based techniques are not able to provide accurate and reliable solutions in a complex indoor environment. As a solution, adequate indoor localization in IoT networks improves the quality of the offered services, providing several key benefits. First, it optimizes the management of the connected objects by effectively monitoring equipment and products. In addition, it ensures more security by facilitating the monitoring of sensitive areas and the detection of attacks. Besides, it makes it easier for users to navigate complex environments, enhancing the user's experience. It also allows customizing location-based services and, thus, meeting the utilizer's specific spatial needs. Moreover, indoor localization facilitates the automation of tasks by identifying the position of objects or people in real time.
To overcome these major challenges, routing and indoor localization problems in IoT networks should be solved effectively. However, the complexity and dynamics of these networks require the application of powerful and innovative paradigms and methodologies. In this context, multi-objective optimization proves to be a promising approach. As it offers the possibility of simultaneously taking into account several performance criteria, this method makes it possible to find optimal solutions in complex environments as IoT networks. Additionally, metaheuristics show high effectiveness in solving the faced real-world problems. Moreover, Deep Learning techniques can deal with complex problems, particularly in IoT, by offering innovative and adaptive solutions and thanks to their ability to extract patterns from data. It is in this context that the contributions of this thesis, presented in the following section, are integrated.

1.2 Research objectives and main contributions

In this thesis, we will take advantage of the two following complementary paradigms: optimization and Deep Learning techniques by combining them to propose innovative solutions for routing and indoor localization in IoT networks.

- First, we carry out an in-depth review of the literature. The state of the art first studies the routing and indoor localization in IoT networks by exploring the key concepts that should be defined to understand these two issues. Then, it focuses on two innovative areas: optimization and Deep Learning. Afterwards, it presents recent research works related to routing in IoT networks, indoor localization, optimization and Deep Learning and compares them. The state of the art aims at providing the basis of the theoretical contributions of this thesis and the obtained experimental findings.

- Second, a mathematical modeling of routing in IoT networks is proposed. It identifies and formalizes the characteristics of the IoT network and the key objectives, the decision variables and the constraints related to the routing problem. This modelling provides a framework for designing and optimizing the routing algorithms in IoT networks by emphasizing the important objectives such as reducing transmission latency, improving delivery rate packets and ensuring efficient energy management. It aims at establishing the basis of developing robust and efficient routing solutions.

- Thirdly, a routing approach, relying on the MOGWO algorithm [2], is introduced in order to model reactive and multi-objective routing in an IoT network having hybrid topology. In this approach, routing is considered as an optimization problem whose objective is to define an efficient route through multiple hops. This approach is applied to attain two major objectives: efficient energy management and transmission latency reduction.

- Forth, a geographic routing procedure, based on an indoor location solution, is suggested. In this procedure, the following approaches are used:

  – We propose an improved version of the multi-objective metaheuristic algorithm MOGWO, that aims at solving efficiently problems having large number of objectives. This new algorithm, called Improved MOGWO (IMOGWO), modifies the fundamental equations of MOGWO to explore more effectively the agent's positions and optimize their updating. Our multi-objective geographic routing approach relies on IMOGWO to select multi-hop routes when transmitting data in the IoT network.

  – We develop an approach to create an optimizer applied to train Deep Learning (DL) models based on a metaheuristic method. This approach consists in modeling the training of a DL model as a single-objective optimization problem. Its main objective is to accelerate the convergence of DL models towards optimal weight values while improving their performance.

  – Then, we suggest an indoor localization approach in an IoT network. This approach relies on the previous contribution since it consists in developing a Deep Learning model optimized and trained using the Gray Wolf Optimizer (GWO) metaheuristic optimization algorithm [3]. The localization process is primarily based on the measurements of Time of flight (ToF) taken from Ultra-Wide Band (UWB) signals between the connected objects and the fixed anchors to train the DL model and estimate the positions of mobile objects.

- Finally, we introduce a hybrid and multi-objective geographic routing approach to be applied in IoT networks having mesh architecture. In fact, the proposed geographic routing requires determining the positions of connected objects to direct messages through the network by using method relying on the previously-described localization approach. Additionally, the proposed routing solution is considered as multi-objective because it applies the developed IMOGWO approach to optimize the routing process by considering various objectives simultaneously. The overall objective of this approach is to ensure efficient message routing while meeting the specific requirements of IoT networks by establishing optimized multi-hop routes. This routing approach aligns with the field of Edge Computing since the construction of routes is distributed between objects in the IoT network.

The proposed approaches are tested and evaluated by using performance indices and conducting real experiments and simulations. The obtained results are compared to those provided by other recent existing solutions applying standard comparisons and inferential statistical tests. They prove that the introduced approaches outperform the existing ones.
The structure of this thesis manuscript is described in the following section.

1.3 Structure of the manuscript

This manuscript is divided into four chapters organized into two distinct parts. The first part, including chapters 2 and 3, presents the state of the art. On the other hand, the second part, made up of 4 and 5, details the proposed contributions.
In chapter 2, we explore routing and indoor localization in IoT networks. We first define IoT by presenting its different network topologies, its communication technologies as well as its various application areas and the main challenges it faces. Then, we introduce the concept of routing in IoT networks, its various types and the performance indices used to evaluate the performance of the routing algorithms. Afterwards, we synthesize recent research on routing in IoT networks. We aim to develop a geographic routing that takes advantage of the positions of the network nodes to optimize the routing process. To this end, we focus, subsequently in this chapter, on indoor localization in IoT networks by defining it and describing its importance. Afterwards, we illustrate the techniques applied to create indoor localization solutions and present the evaluation criteria utilized in these solutions. Finally, we compare and analyze recent studies on indoor localization in IoT networks.
Chapter 3 deals with optimization and Deep Learning techniques. In the first part, the different types of optimization problems, their fundamental concepts, the main resolution methods and the

used evaluation criteria are presented. Then, a review of recent techniques applied to solve these optimization problems is provided. The second part of this chapter focuses on DL by enumerating its main areas of application, particularly in regression. We also depict the theoretical foundations of DL, the main DL techniques employed to solve regression problems and the performance criteria employed to evaluate their performance. After that, this chapter provides a detailed review comparing recent studies on solving the routing problem in IoT networks using optimization algorithms. Finally, it examines recent research works focusing on the resolution of the indoor localization problem in IoT networks by applying metaheuristics and/or DL techniques.

Chapter 4 offers an in-depth description of our theoretical contributions. It begins with a mathematical modeling of routing in IoT networks. Next, we present an improvement of the existing metaheuristic MOGWO optimization algorithm, named Improved MOGWO (IMOGWO). Subsequently, we detail an approach for optimizing the parameters of DL models using metaheuristics. In the following section, we describe our proposal for indoor localization in IoT networks, based on the previously presented approach. This localization solution will allow us to develop a geographic and hybrid routing algorithm, designed specifically for IoT networks with mesh architecture. The last section of this chapter is dedicated to the detailed presentation of this routing proposal.

The last chapter, 5, illustrates the experiments and simulations carried out to evaluate the presented contributions. It also analyzes the obtained results. The suggested approach for reactive and multi-objective routing based on MOGWO is evaluated through real experiments detailed and presented in the first section. The obtained results are also presented and analyzed to highlight the good performance of the proposed approach. In the second section, the performance of the developed IMOGWO algorithm is theoretically evaluated by solving the DTLZ benchmark problems [4]. The results provided by IMOGWO and those given by other widely-used optimization algorithms are analyzed and compared. In the third section, we describe the tests carried out to evaluate the effectiveness of our contributions: the optimization of the weights of the DL models utilizing metaheuristic algorithms and the development of a DL model for indoor localization based on UWB ToF measurements. The evaluation of the findings provided by the introduced approach are compared and analyzed to those obtained by other indoor localization solutions. In the last section, we illustrate the experimental results given by the proposed geographic and hybrid routing relying on IMOGWO. These tests were conducted: to evaluate the performance of the suggested routing solution; compare it with those of other existing routing solutions and to validate the effectiveness of the IMOGWO algorithm when applied in a routing context, apart from the DTLZ reference problems.

Finally, we present our conclusions and perspectives as well as future directions of our future research study.

Part I

State of the Art

Chapter 2

Routing and indoor localization in IoT networks

Contents

2.1 Introduction

The first chapter of this thesis report focuses on routing and indoor location in IoT networks. We will define IoT by presenting its network topologies, its communication technologies, its application domains and the major challenges it faces. Then, we will explore in details two fundamental aspects of IoT networks: routing (particularly geographic routing) and indoor localization. First, we will define the routing in an IoT network, explore its different types as well as the relevant performance indicators and present a review of the recent studies dealing with the former. Afterwards, we will define indoor localization, discuss its importance and enumerate the various existing localization techniques as well as the relevant evaluation criteria. Finally, we compare and review recent studies on indoor location in IoT networks, providing a comprehensive overview of these two important areas.

2.2 Introduction to the IoT

In recent years, IoT has been considered as a major technological advancement that has revolutionized our interaction with the physical world by connecting numerous smart devices and objects within a dedicated interconnected network. These objects, equipped with sensors and actuators, have become intelligent entities that can collect, share and process data autonomously, which has led to the emergence of innovative applications in various fields such as health, industry and home automation. IoT has created a dynamic ecosystem where interactions between the connected entities surpasses simple data transfer, showing its remarkable impact on the way we live, work and interact with our surrounding environment. Thanks to its several advantages, an IoT network is increasingly integrated into different application areas. For this reason, the number of connected IoT devices used worldwide is increasing exponentially to reach more than 75 billion devices in 2025 [1] (Figure 2.1). According to [5], approximately 500 billion of devices equipped with sensors will be connected to the Internet by 2030. As the number of connected IoT objects increases, IoT networks become more complex, requiring the deployment and organization of their components according to a specific topology. The following section lists the most common topologies of IoT networks.

2.2.1 Topologies of the IoT networks

The components of the IoT network are structured according to a given topology whose choice is one of the factors that influence the good functioning and efficiency of the network. In fact, several network topologies were introduced in the literature. The most common ones are stated below:

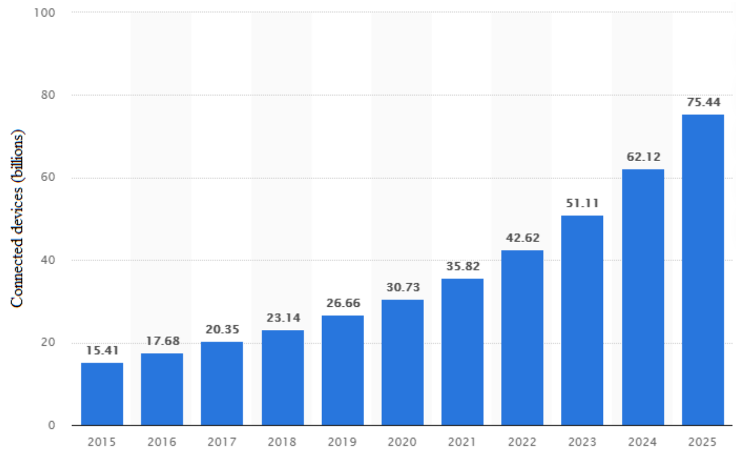


Figure 2.1: Connected IoT devices from 2015 to 2025 [1]

- Point-to-point topology (Figure 1(a)) ensures direct communications between two communicating nodes;
- Star topology (Figure 2.2(b)) allows communications only between the terminal nodes and the central point;
- Mesh topology (Figure 2.2(d)): in this topology, nodes are fully interconnected, allowing data propagation through a multi-hop architecture;
- Hybrid topology (Figure 2.2(c)) consists in connecting a set of star networks in a mesh network;
- Tree topology (Figure 2.2(e)) where the components of the network are organized hierarchically, i.e., the nodes of each level are connected to the nodes of lower level.

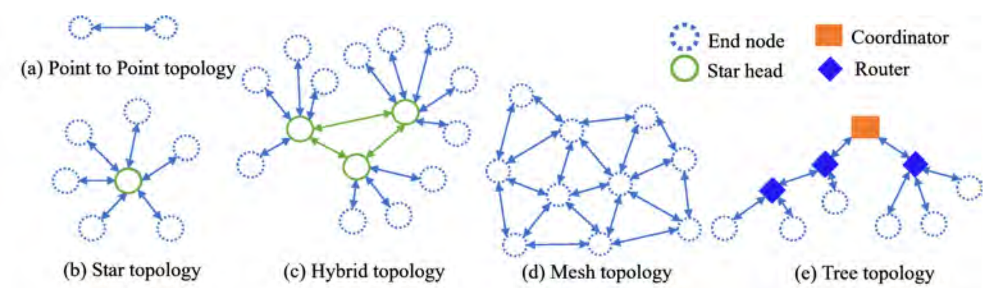


Figure 2.2: The most common topologies for IoT networks

Data in IoT environments must be managed effectively to ensure the IoT system success through their reliable and rapid processing and transmission. In such environments, communication technologies play a central role in data efficient transmission. They are continually evolving to meet the changing needs of IoT by taking advantage of the achieved technological advances, particularly in wireless communications. The next section presents the communication technologies used in IoT networks.

2.2.2 Communication technologies used in the IoT networks

In fact, connectivity ensures a well-organized and efficient IoT environments. However, no universal communication solution that can be applied in all use cases and all IoT applications was proposed in the literature as IoT networks employ a variety of communication technologies, including both wired connections (e.g., Ethernet, HomeGrid/G.hn and Multimedia over Coax Alliance (MoCA)), which make communications more stable and reliable, and wireless technologies (e.g., Wi-Fi, Bluetooth and cellular networks) that guarantee flexible and mobile connectivity. The choice of the adequate technology to be used in a specific IoT application is determined according to the specific needs of the latter, allowing them to meet various requirements of different communication environments. Figure 2.3 presents the classification of the communication technologies utilized in IoT networks and some illustrative examples of these technologies.

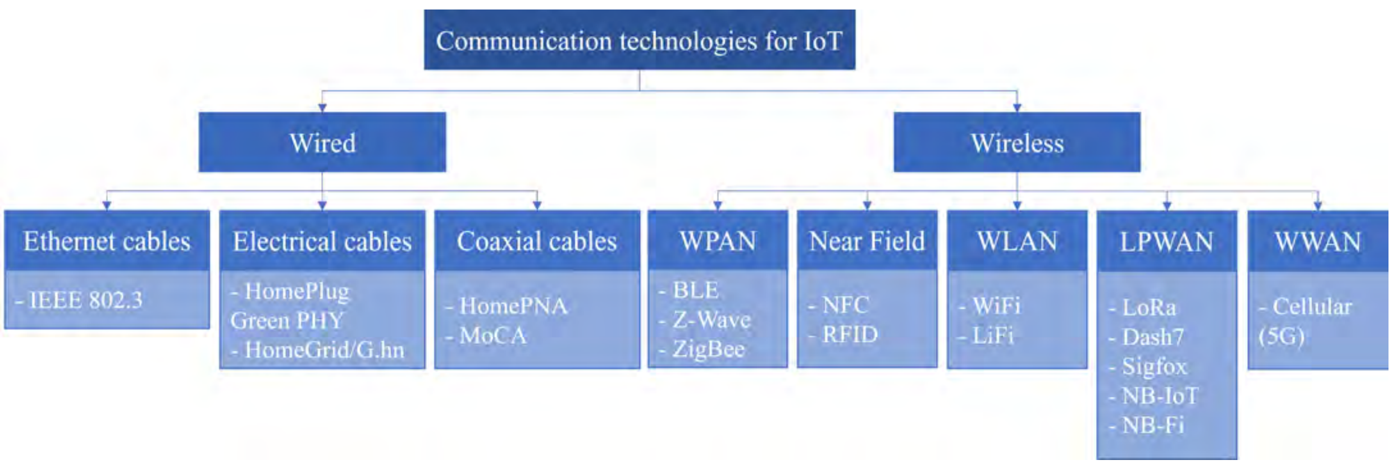


Figure 2.3: The communication technologies used in IoT networks

Numerous technologies and associated protocols were introduced in the literature and several attempts were also made to standardize these technologies and make them interoperable, such as MATTER [6]. In the last decades, communication technologies have continually evolved to satisfy the new and changing IoT requirements, which renders IoT applications more efficient. Thanks to this unceasing development, a wide range of IoT applications have been designed and implemented, affecting man's personal and professional live. The following section presents the classification of IoT application areas and lists some examples of these domains.

2.2.3 Application domains of the IoT

IoT is employed in different areas because of its advantages such as real-time data collection, remote monitoring and process optimization. It is also utilized in many domains to improve their operational efficiency, reduce services costs and meet the growing user needs. Figure 2.4 presents a classification of IoT application domains.

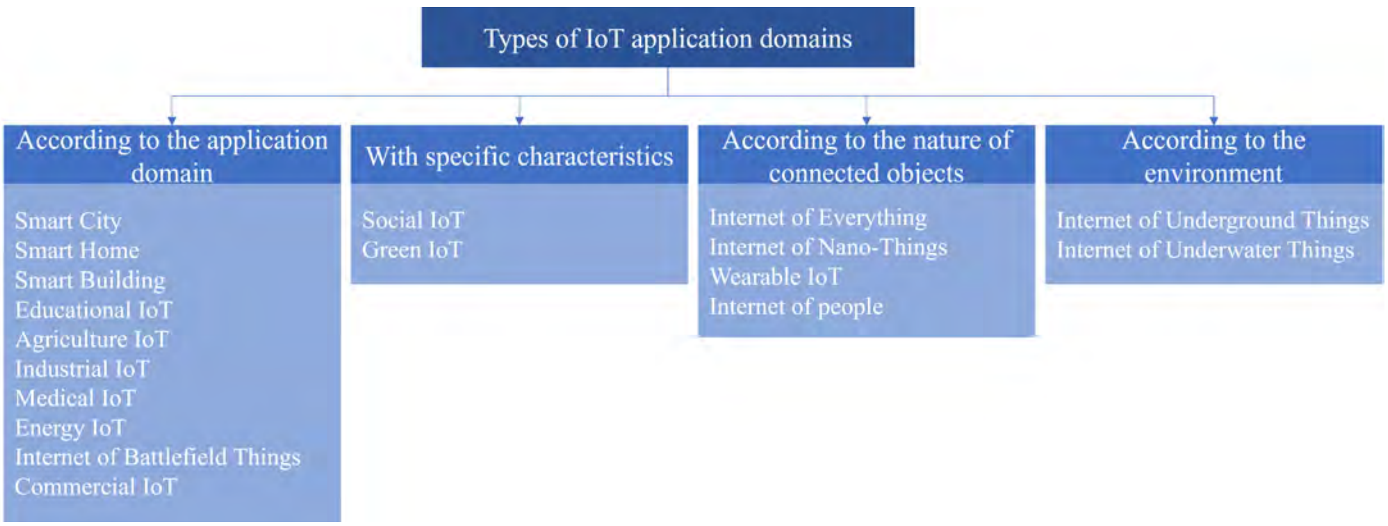


Figure 2.4: The main application domains of IoT

Although IoT is continuously developing and expanding in various application domains, it still faces many challenges and problems. The main challenges of IoT are presented in the following section.

2.2.4 Challenges of the IoT

The major challenges of the IoT applications are:

- Indoor localization: consists in determining the position of the objects inside buildings or enclosed spaces;
- Routing: is to select the optimal paths in order to route data efficiently, particularly between the network wireless nodes which participate in the routing of information on the meshed part of the wireless topology;
- Effective data management: it is ensured by intelligent data aggregation, fast and reliable data transmission, rapid processing and consideration of the various managed data;
- The security and confidentiality of the transmitted and stored data;
- The consideration of the heterogeneity of the used technologies;
- The scalability of the IoT networks and the scalability of the infrastructures;

- Optimization of energy consumption;
- Operational safety as well as stable and resilient connectivity;
- An increased response to the growing consumer's expectations.

This section presents an overview about communication technologies, their application areas and the main challenges of IoT described in our publication [7]. The following section deals with routing in an IoT network and highlights its importance in ensuring smooth communication.

2.3 Routing in The IoT networks

In the context of IoT networks, routing refers to the process of choosing routes taken to transfer data between the connected devices (e.g., sensors, actuators and gateways) across the network. Unlike the traditional communication networks, where transmission paths are often static and pre-established, routing in IoT networks is dynamic and changing because of the varying nature of the devices and the environment in which they are used. This dynamism requires flexible routing mechanisms that can be adjusted according to the changes in the network conditions such as node availability, topology variations and resource constraints. Due to the direct impact of routing on the performance and reliability of IoT applications, an efficient routing is of crucial importance in IoT networks. Various types of routing are available in order to meet the requirements of different IoT environments. The following subsection aims to describe the principles, advantages and disadvantages of the main types of routing.

2.3.1 Types of routing

Routing protocols applied in IoT networks can be classified according to several criteria stated below.

2.3.1.1 According to the network topology

a **Flat routing**: it is characterized by a network structure where all objects are considered equivalent. Unlike other more hierarchical routing scheme, in flat routing, each IoT object assumes the same roles and functionalities. These objects collaborate to accomplish the overall network task by sending data directly to the sink node via a multi-hop communication mode (Figure 2.5).

b **Hierarchical routing**: it is based on the hierarchy where transmissions take place on, at least, two levels. In this protocol, nodes are divided into multiple sub-groups, which facilitates the organization the network management. In such routing, star topology is the most commonly

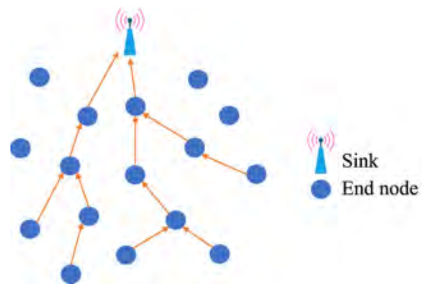


Figure 2.5: Flat routing

used (Figure 2.6). IoT objects are grouped into clusters; each of which is supervised by a node or "cluster head". The latter is responsible for aggregating data provided by from cluster member objects before sending them to the final destination. Hierarchical routing can also be achieved by a tree topology where routers are organized into different levels. The routers of the lower level forward the traffic to higher-level routers to be routed towards other parts of the network.

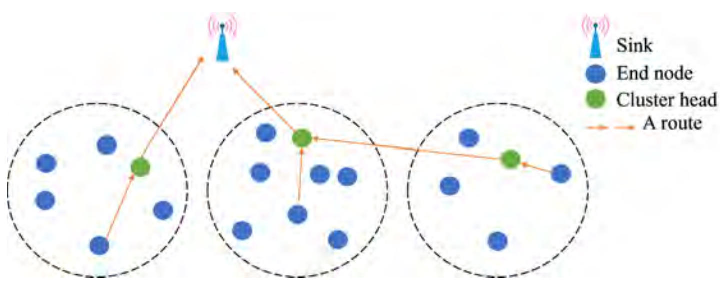


Figure 2.6: Hierarchical routing in a star topology

c **Geographic routing**: it uses geographical location information of devices to optimize data transmissions. This type of routing relies on the estimation or calculation of the distance between two objects according to their spatial coordinates, which makes it possible to determine the shortest or, more generally, the most efficient path to route data.

d **Direct routing**: it is a communication scheme where data is transferred directly from one node to another without an intermediate relay, as it is the case of a fully-meshed topology (Figure 2.7(a)). In other scenarios, objects can be located within a single hop distance of the base station or that of a central node in direct routing (Figure 2.7(b)). Unlike more complex routing schemes which involve relays or intermediate nodes, direct routing simplifies the network structure by establishing direct communications between objects and with the base station.

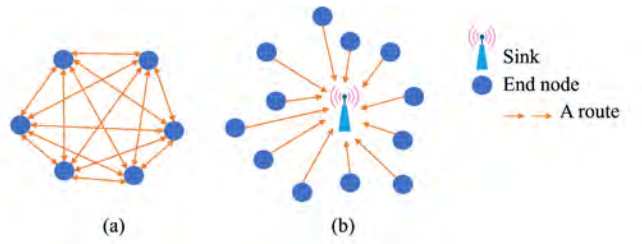


Figure 2.7: Direct routing

2.3.1.2 According to the moment at which the routing path is determined

a **Proactive routing**: it is a communications management strategy where transmission paths are established in advance and are ready to be used when needed. Proactive routing is employed to anticipate the optimal paths between the network nodes and configure them in advance in the routing tables of IoT devices. Due to the variations of the topology between the connected objects caused by the movement of nodes and changes of the radio propagation conditions between them, this type of routing is not considered efficient.

b **Reactive routing**: it is a method of managing communications where routes are established only when needed and they are removed once the delivery of data to the destination is accomplished. Unlike proactive routing that preconfigures routes, reactive routing satisfies dynamically data routing requirements by finding and constructing routes when data is sent.

c **Hybrid routing**: is an approach that combines elements of reactive routing and those of proactive routing to optimize communications. This technique benefits from the advantages of both routing paradigms (proactive and reactive) as a function of the specific network needs and the changing conditions of the IoT environment.

2.3.1.3 According to the functioning of the used protocol

a **Negotiation-based routing**: it is an approach used to manage communications based on the exchange of data descriptors between the network nodes before the actual transmission of data. This approach aims at reducing information redundancy by allowing nodes to negotiate transmission details before sending data to destination. In fact, data descriptors include information about the transmission requirements such as Quality of Service (QoS), bandwidth constraints and routing preferences.

b **Multi-path routing**: it is a communications management strategy which consists in establishing several paths leading to the desired destinations. This approach is generally applied to improve network reliability and resilience by providing alternative routes in the case of failure or congestion on the main path. The extreme example of establishing several paths is the flooding of the entire network, which is very costly in terms of the consumed energy and resources as well as in terms of the used bandwidth. Having several possible paths, including at least one emergency path, is an interesting option.

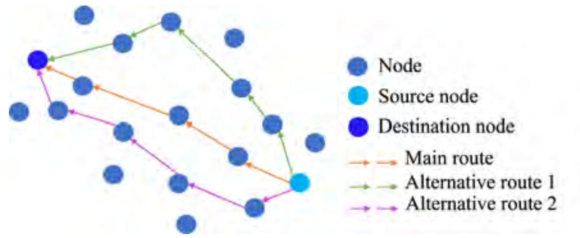


Figure 2.8: Multi-path routing

c **QoS-based routing**: it is a communications management approach employed to meet different quality of service requirements such as transmission delays, reliability level and bandwidth. This routing method ensures the routing of data according to predefined performance criteria to satisfy the specific needs of IoT applications.

d **Non-coherent routing**: in this communications management approach, network nodes process locally raw data before transmitting them to nodes responsible for their subsequent processing. Non-coherent routing allows the efficient distribution of data processing as a function of the specialization of the network nodes.

e **Coherent routing**: In this strategy, sensor objects perform the minimum local processing of raw data through several processes (e.g., adding the date, clocking the time or removing duplicates) before transmitting these processed data to the specialized objects, often called aggregators, to be further processed.

f **Data-oriented routing**: it is a two-step process (Figure 2.9). In the first stage, the object transmits its interests in the network by specifying the data it will retrieve. Then, the sensor objects respond to the requests by sending the required data through the reverse path of the request to the requesting object.

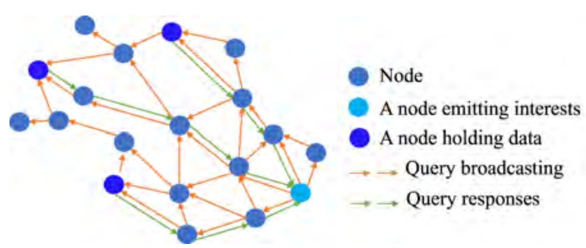


Figure 2.9: Data-oriented routing

2.3.2 Synthesis

Table 2.1 summarizes the principles, advantages and disadvantages of each type of routing.

Table 2.1: Classification of routing protocols

Criterion	Type	Description	Advantages	Disadvantages
Network topology	Flat	Consider all nodes in the network as equivalent, allowing direct communication without routing hierarchy.	Simplicity, low latency, scalability, flexibility	Limited scalability, lack of redundancy, vulnerability to attacks, higher energy consumption
	Hierarchical	Involve dividing the network into levels of clusters where each cluster is managed by a central node called the cluster head.	Scalability, redundancy, better fault tolerance, simplified management	Complexity, increased latency, cluster head overload
	Geographic	Use location information to determine paths between communicating nodes.	Reduced latency, adaptability to topology changes and node movements, minimal information exchange except for neighbor location within one hop, relatively low memory cost, ideal scalability, simple addressing, easy network join, locality	Need for precise location, location cost, vulnerability to attacks
	Direct	Route sensor data from sensor objects to a base station in a single hop.	Low latency, simplicity, energy efficiency	Limited range, vulnerability to failures, density constraints
Path determination timing	Proactive	Maintain pre-defined communication paths between IoT network nodes before communication requests occur.	Low latency, proactive management, stability	Control overhead, resource consumption, lack of flexibility
	Reactive	Establish communication routes only in response to communication requests.	Efficient resource utilization, adaptability, scalability	Increased latency, instability in dynamic networks, control overhead, excessive resource usage
	Hybrid	Combine proactive and reactive approaches to establish communication routes.	Flexibility, balance between performance and resource usage, stability	Complexity in coordinating both routing methods, resource consumption
Protocol operation	Negotiation-based	Use data descriptors to exchange a series of negotiation messages before actual data transmission.	Elimination of redundant information, adaptability to specific application requirements, reduced network load	Complexity, increased latency, need for node synchronization
	Multi-path	Establish multiple communication paths to a destination.	Increased fault tolerance, load balancing, improved performance	Complexity (additional coordination and maintenance), control overhead, implementation cost
	QoS-based	Adjust communication paths based on QoS criteria.	Satisfaction of specific performance requirements, efficient resource utilization, improved reliability	Complexity (monitoring QoS parameters), control overhead, need for QoS-awareness in nodes
	Non-coherent	Involve local processing of raw data by objects before transmission to specialized nodes for further processing.	Reduced network load, node autonomy, latency reduction	Loss of context or details in transmitted information, redundancy risk (no coordination between nodes), limitations of local processing by node capabilities
	Coherent	Involve minimal initial processing of data by objects.	Context preservation, efficient resource usage, analysis flexibility	Increased latency, need for coordination, risk of overload
	Data-oriented	Respond to specific data requests by broadcasting interests in the network.	Targeted resource usage, adaptability to changing network needs, reduced overhead	Potential latency, need for coordination between interest-emitting nodes and data-holding nodes, risk of data loss

2.3.3 Comparaison and review of the recent studies dealing with routing in IoT networks

In the literature, many studies focused on solving the routing problem in IoT networks. Table 2.2 presents a comparison between the recent approaches introduced to address this issue.

2.3.4 Synthesis

In this section, we introduced the notion of routing in an IoT network, its various types and the major criteria used to measure the performance and efficiency of the routing protocols. We also described the main recently-proposed routing solutions. In our publication [8], we studied routing in IoT environments. The evaluation of these research works demonstrates that each of them focused on a limited number of needs and performance criteria. Additionally, most of them examine a specific type of IoT environment, which limits their effectiveness in addressing challenges faced in other IoT contexts. Additionally, the majority of the previous studies used simulations to illustrate the improvement of the routing performance. However, they did not employ a simulator specific to IoT networks and, much less, on real testbeds. Therefore, we propose, in the thesis report, a routing algorithm that takes into account a wider range of objectives which can be adapted to IoT environments and allow improving the conditions of experimenting and evaluating the routing solution.
Among the various routing types, we are particularly interested in geographic routing due to its advantages as an attractive routing solution used in IoT networks. This routing was studied in [9] and [10] where authors recommended its employment as a key solution applied to transmit information by overcoming the scalability and mobility challenges.
To implement a geographic routing algorithm in an IoT network, it is essential to adopt a localization solution because geographic routing utilize location information to establish the paths between the communicating nodes. The following section is devoted to indoor localization in IoT environments.

Table 2.2: Comparison and review of recent works addressing the routing problem in IoT networks

Paper	Year	Type	Approach/Objectives	Techniques	Results	Disadvantages
[11]	2023	hierarchical	energy efficiency and security in a dynamic topology	selection of cluster heads based on sink node announcements to neighboring nodes, compression with Huffman coding, and encryption with RSA asymmetric algorithm	energy consumption: 1% of initial energy security percentage: 93%	No real-world experiments; Simulations performed using Matlab instead of a specific IoT network simulator; Simulated network limited to 5 nodes.
[12]	2023	proactive, QoS-based	Consider current and future state of the network and ensure differentiated quality of service support.	Graph Neural Network and Reinforcement Learning	PDR: 99% Throughput: 60 packets/sec Average latency: 60ms	Routing strategy not able to adapt efficiently to topological changes as it requires retraining of the model used in such situations; No real-world experiments.
[13]	2023	negotiation-based	Model the vehicle routing problem as a Coalitional bargaining game [14]	Multi-agent reinforcement learning	88% reduction in execution time Average accuracy: 77% Average optimality gap: 3.9%	Limited to 3 agents; Proposed algorithm not evaluated against known routing metrics and not compared with other routing algorithms; No real-world experiments.
[15]	2023	QoS-based	Minimize energy consumption and improve routing security	Ant Colony Optimization (ACO)	Latency $\leq$ 14.5ms Over 9.31% improvement in energy consumption compared to other compared algorithms	No mobile nodes; No heterogeneous nodes; No real-world experiments.
[16]	2023	coherent, QoS-based	Develop a routing mechanism for multi-hop LoRa networks considering QoS requirements	Versatile LoRaute node acting as access point and base station	PDR: 96.81%	Performance of proposed strategy not evaluated using standard routing metrics and not compared to other routing solutions based on those criteria.
[17]	2023	data-driven	Divide data based on content and adjust routing topology based on content	Binary Gray Wolf Optimization algorithm and fuzzy logic	PDR: 98.35% Throughput: 5700kbps	No data redundancy handling; No real-world experiments.
[18]	2022	hierarchical	a version of Hierarchical Smart Routing Protocol (HSRP) [19] integrating a new energy management function	Smart Slumber: put cluster heads and member nodes to sleep after each cycle for a specified duration	PDR: >92% Network lifetime: 7000 cycles	No real-world experiments.
[20]	2022	flat, multi-path	A secure version of AODV [21] against Blackhole attacks	Establish multiple routes from source to destination to bypass attack paths	PDR: 85% Average latency: 220ms Overhead: 60% Throughput: 2.2 packets/sec	Performance of proposal only compared to AODV, limiting comprehensive evaluation of its effectiveness; No real-world experiments; No mathematical modeling presented.

Continued on next page

Table 2.2 – continued from previous page

Paper	Year	Type	Approach/Objectives	Techniques	Results	Disadvantages
[22]	2022	proactive	Develop a routing protocol suitable for Large Regions of Interest (ROI) monitoring and ensuring balanced distribution of transmission power load	Joint Decentralized Antenna (JDA) algorithm	25% improvement in network lifetime	Results proving effectiveness of proposal and comparison with other solutions not fully presented.
[23]	2022	geographic	Optimize routing in WSNs for multimedia applications by prioritizing high-throughput paths over shorter paths to base stations	Avoid persistent routing through the same path and implement network void zone bypass	Average latency: 5ms Packet Delivery Ratio (PDR): 97.25% Residual energy: 99.7%	Proposal not evaluated in terms of throughput, which is a key criterion for Multimedia networks; No real-world experiments.
[24]	2021	geographic	improve the efficiency of geographic routing in urban Intelligent Vehicle Networks (VANETs)	use a Link Dynamic Behavior (LDB) model to evaluate link quality and select the most reliable paths	PDR: 80%-98% Latency: 0.4s-0.7s Throughput: 0.08Mbps-0.097Mbps	No real experiments
[25]	2021	QoS-based	achieve efficient routing ensuring QoS in Software-defined Networks (SDNs)	various Machine Learning (ML) algorithms and a new classification method called MACCA2-RF&RF	average throughput: 208.07kbps average latency: 106.52ms PDR: 99.48%	No mobile nodes; No real experiments
[26]	2020	geographic	reduce routing overhead by maintaining information on a single neighbor for each node and effectively balance node energy consumption	apply a mean square error algorithm to solve object localization problem	PDR: 50% Delay: 3.6msec-5.5msec Average hop count: 3.25 hops	Cumulative computation time problem to route data to destination; Comparison with only one other routing algorithm; No real experiments
[27]	2020	flat	an enhanced version of Ad hoc On-Demand Distance Vector (AODV)	exploit the relationship between transmission range and density to dynamically adjust transmission power usage	PDR: 62% Average latency: 730ms Jitter: 0.063sec Throughput: 150000 kbits/sec	Comparison of proposal's performance only with AODV, which does not fully evaluate the efficiency of the proposed approach; No real experiments.

2.4 Indoor localization in IoT networks

In the continuously-developing IoT applications requiring precise localization of outdoor environments, GNSS are commonly used to provide real-time outdoor location. However, due to their poor ability to penetrate indoor environments, it is essential to develop alternative solutions to meet the needs for precise localization in such environments within IoT networks. Indoor localization refers to the ability to accurately determine the position of objects and devices inside buildings or in confined spaces. It is one of the major challenges that faces IoT deployments due to the complexity of indoor environments. In fact, the latter present physical obstacles such as walls, ceilings and floors, which hinder the transmission of signals needed to accurately locate devices. Therefore, developing efficient and adequate indoor location solutions is important to ensure that IoT applications function properly in these environments.

2.4.1 Indoor localization techniques

Indoor location solutions can be classified into two categories: range-based and range-free solutions. In the remainder of this subsection, we will define and list the common indoor localization techniques.

2.4.1.1 Range-free localization techniques

Range-free localization techniques are characterized by their ability to estimate the position of an object without using measurement equipment since they do not require distance or angle measurements between nodes. They generally apply collaborative approaches to derive the estimated positions from the information about the neighborhood and the geometric methods. The major advantage of these techniques consists in the fact that they can operate without using specific hardware infrastructures, which makes them less expensive. We present, below, examples of the most widely-employed range-free techniques.

(a) **Centroid**: The Centroid algorithm, proposed by [28], is generally applied to estimate the position of a node based on the information about the neighborhood. An example of the Centroid application is shown in Figure 2.10. It relies on the following steps:

   (i) Collecting information about the neighborhood: Let N be the number of neighbors of the node to be located. Each neighbor i transmits its estimated position (x_i, y_i) to the node to be localized. Neighboring nodes with known positions serve as anchors, providing essential reference points for locating the target node.

(ii) Calculating the estimated position $((\hat{x}, \hat{y}))$ by applying equation 2.1:

$$\hat{x} = \frac{1}{N}\sum_{i=1}^{N} x_i, \, \hat{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{2.1}$$

The Centroid algorithm is relatively simple and does not require the measurement of the distance between nodes. However, its accuracy depends on the density of the anchors in the neighborhood as well as on the spatial distribution of nodes in the network.

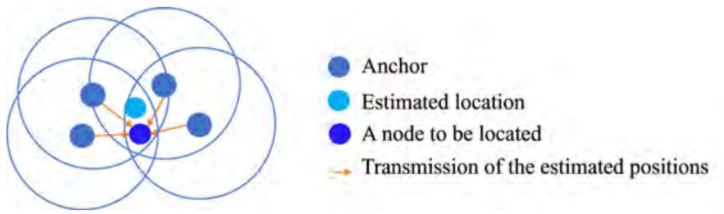


Figure 2.10: Estimation of the location obtained by the Centroid algorithm

(b) **Distance Vector-Hop (DV-HoP)**: The DV-HoP algorithm, introduced by [29], estimates the position of a node based on the distance separating the nodes from the anchors whose positions are known. Indeed, the distance between a node and an anchor can be approximated according to the number of hops necessary to reach the anchor. DV-HoP relies on the following steps:

(i) The anchors diffuse their locations and the number of hops (incremented with each intermediate hop) throughout the network;

(ii) Each receiver node i keeps H_{ij}, the minimum received number of hops, and uses it as the shortest distance to the anchor j;

(iii) The mean distance per hop, d_{avg}, is calculated by the anchors. It is transmitted to the neighboring normal nodes. It can be computed by dividing the distance between two anchors by the number of hops.

(iv) From the positions of the anchors, the values H_{ij} and d_{avg}, a normal node can calculate its position. The trilateration method can be applied, for example, to estimate the position of node i. It is described in (a).

The DV-HoP algorithm is a cost-effective and adaptable localization method used in wireless networks that do not require the employment of specific hardware to take measurements. Although it is efficiently employed in dense networks, its accuracy can be compromised by the noise effect and the measurement errors, especially in less dense environments. Its performance is also influenced by the density and spatial arrangement of the anchors as well as by the topological stability of the network.

Many variants of DV-HoP have been proposed by the scientific community. The most recent ones are: CMWN-DVHop [30], WRCDV-Hop [31] and HW-DV-HopCSO [32].

(c) **Adaptive Point-In-Triangulation (APIT)**: The APIT algorithm, developed by [33], combines the adaptive approach with the Point-In-Triangulation (PIT) test to reduce the estimated region of nodes to be localized, which allows approximating their position without indicating the distance between nodes. The APIT algorithm is based on the principles stated below:

(i) Let A be the set of anchors in the wireless network. The positions of these anchors are known. Each normal node i collects beacon messages from the neighboring anchors;

(ii) The APIT algorithm reduces the estimated region R_i of the normal node i by conducting the PIT test using different combinations of the neighboring anchor points. This test is represented in Figure 2.11 and equation 2.2:

$$PIT_{ij} = \begin{cases} 1, & \text{if node } i \text{ is inside the triangle formed by the anchors } j \in A \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

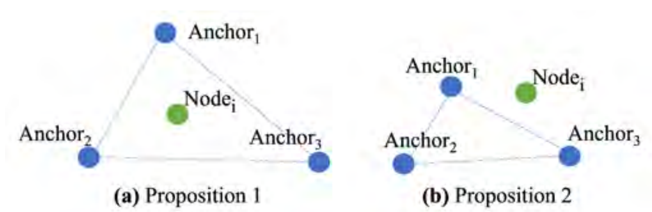


Figure 2.11: The Point-In-Triangulation test

(iii) Grid-scanning algorithm: APIT uses a grid-scanning algorithm to derive the region of the intersection of all triangles formed by the anchor points (Figure 2.12). The intersection region is represented by equation 2.3:

$$\text{Region}_i = \bigcap_{j \in A} \{PIT_{ij} = 1\} \tag{2.3}$$

(iv) The center of the intersection region Region_i is considered as the estimated location $(\hat{x}_i, \hat{y}_i)$ (Figure 2.12).

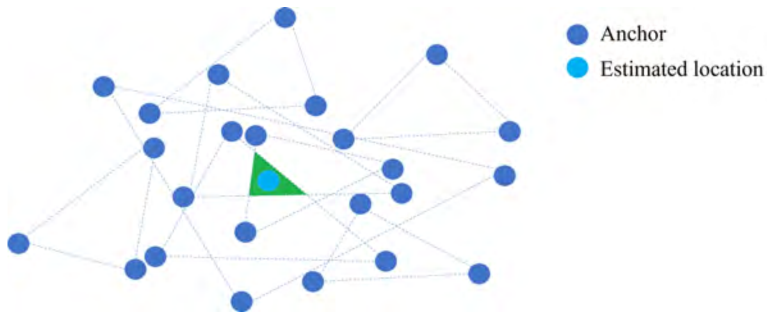


Figure 2.12: APIT algorithm

Despite its high accuracy in environments without distance indications, APIT suffer from memory and computational resource constraints due to the need to divide the network area into grids, which limits its use in IoT networks.

(d) **Convex Position Estimation (CPE)** : The CPE algorithm, suggested by [34], is applied to estimate the position of a normal node by utilizing connectivity constraints and its neighboring anchors to define an overlapping region. It is also employed to calculate the center of the estimated rectangle of this region to obtain the estimated position of the normal node. CPE algorithm consists of the following steps:

(i) Initialization: Let P be the set of anchors whose positions are known. Each anchor p_i is characterized by its coordinates (x_i, y_i);

(ii) Constructing connectivity matrix C where $C_{ij} = 1$ if the normal node can communicate with the anchor j. Otherwise, $C_{ij} = 0$;

(iii) The CPE algorithm uses connectivity constraints to estimate the overlap region between the communication regions of neighboring anchors;

(iv) The CPE algorithm defines the Estimated Rectangle (ER) which bounds the overlap region (Figure 2.13). The four sides of the ER are parallel to the x-axis and the y-axis;

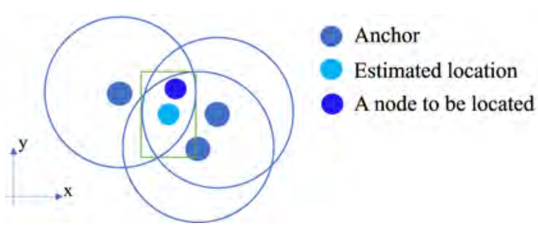


Figure 2.13: CPE algorithm

(v) The estimated position of the normal node is given by the center of the estimated rectangle. The latter is calculated by considering the mean of the coordinates of the neighboring anchors taken into account to determine the overlap region.

Although the CPE algorithm is efficient and simple to implement, it has some limitations, such as connectivity dependence, computational complexity and communication overhead, when applied in a wireless network.

2.4.1.2 Range-based localization techniques

Range-based localization techniques utilize the environmental measurements in the context of IoT to estimate the distance or angle between devices, which allows estimating the positions of the nodes. Their implementation generally requires the use of more expensive hardware and may involve more complex deployment of equipment due to the need to calibrate and maintain measurement tools. However, these techniques are efficiently employed in IoT applications requiring high precision, such as the environmental monitoring of precision or navigation inside buildings. The commonly-used measurement methods [35] are described below:

(a) **Received Signal Strength Indicator (RSSI)** [36]: is the measurement of the received signal strength during wireless transmissions. It is applied to estimate the distance between the transmitter and the receiver. It does not require the use of any additional equipment and it is intensively utilized in indoor localization. However, its accuracy may be affected by obstacles and non-omnidirectional antennas.

(b) **Channel State Information (CSI)**: combines information about the state of the communication channel describing the propagation of the signal sent by the transmitter to the receiver. According to [37], CSI offers better detection accuracy than RSSI although it is less efficient than the radio time-of-flight (ToF) measurement technique. It is well adopted in Wi-Fi links to which radio transceivers transmit CSI information.

(c) **Angle of arrival (AoA)**: measures the angle at which the signal reaches the receiver. It is a recent technique mainly used by UWB multi-antenna radio tranceivers and more recently utilized by Bluetooth in IoT networks.

(d) **ToF**: measures the time spent by the signal to reach its destination. Using the speed of propagation of the radio signal, this measurement can estimate the distance between two objects. UWB signals enable more precise ToF measurements used in localization solutions. To avoid the existence of a common timepiece between the transmitter and the receiver, the Two-Way Ranging (TWR) protocol [38] is often used to measure the ToF.

(e) **Time Difference of Arrival (TDoA)**: is a variant of ToF used to determine the difference in the arrival times of a signal forwarded by a transmitter towards, at least, two distinct receivers considered as reference points. By combining this measurement with the speed of transmitting the signal, the difference in the distance separating the transmitter and the two receivers can be calculated as demonstrated in [39]. By knowing the positions of 3 anchor receivers, it is possible to evaluate the position of the mobile transmitter.

(f) **Signal-to-Noise Ratio (SNR)** : measures the quality of transmission by comparing the power of a signal to the noise level. It is defined as the ratio between the power of the desired output signal and that of noise. This ratio was expressed in decibels [40].

After these measurements, localization algorithms must, then, be implemented. They are generally based on the trilateration or triangulation techniques applied to determine the position of an object or a device by measuring the distances or angles from several known reference points. The most commonly used localization methods are:

(a) **Trilateration** : This technique measures the distance between nodes to determine the positions by utilizing trilateration algorithms such as the circles method or the hyperbols technique. For example, by applying the former, one of the previously-mentioned measurements is used to determine the position of the object by the intersection of the circles (or 3D spheres) centered on the anchors where the radii are specified by distance measurements. Only three and four measurements are used in 2D and in 3D environments, respectively. A 2D example is shown in Figure 2.14. If the coordinates of the 3 anchors are, respectively, $(x1, y1)$, $(x2, y2)$ and $(x3, y3)$, while r_1, r_2 and r_3 are the radii of the three circles, the equation which calculates the estimated position of the object to be located is the system of non-linear equations resulting from these three circle equations (equation 2.4).

$$\begin{cases} (\hat{x} - x_1)^2 + (\hat{y} - y_1)^2 = r_1^2 \\ (\hat{x} - x_2)^2 + (\hat{y} - y_2)^2 = r_2^2 \\ (\hat{x} - x_3)^2 + (\hat{y} - y_3)^2 = r_3^2 \end{cases} \tag{2.4}$$

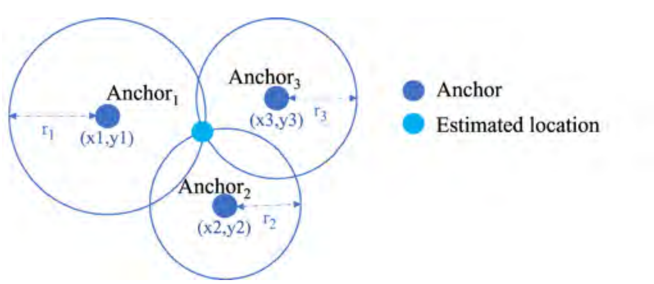


Figure 2.14: The principle of trilateration

The trilateration localization method is a simple and practical approach based on measuring the distances between the anchors and the object to be localized, which makes it efficiently applied in various environments. Despite its high performance, this technique can be sensitive to measurement errors and obstacles. It also requires high number of anchors to ensure adequate coverage of the studied environment, which increases, subsequently, the deployment costs and limits its precision when utilized in some environments. Moreover, the theoretical calculation of positions is often impossible due to small distance errors in reality. To solve these problems, many algorithms (MinMax [41] being the simplest) were proposed in the literature.

(b) **Multi-lateration**: Although this technique is almost similar to trilateration, it employs more than three distance measurements to improve the localization precision. Consider an example of multi-lateration with hyperbolas using n anchors. The coordinates of the anchors are (x_i, y_i) for $i = 1, 2, \ldots, n$. The distance measured between the object to be localized and each anchor is noted by d_i. The reference distance or reference flight time is d_{ref}. Reference measurements are usually determined during the installation or configuration phase of the localization system. They serve as a reference to calibrate the distance measurements or times of flight obtained from the signals emitted by the anchors towards the object. Using the distances measurements, the equations of hyperbolas can be written as follows (equation 2.5):

$$\begin{cases} \left|\sqrt{(x-x_1)^2+(y-y_1)^2}-(d_1-d_{\text{ref}})\right| = c_1 \\ \left|\sqrt{(x-x_2)^2+(y-y_2)^2}-(d_2-d_{\text{ref}})\right| = c_2 \\ \vdots \\ \left|\sqrt{(x-x_n)^2+(y-y_n)^2}-(d_n-d_{\text{ref}})\right| = c_n \end{cases} \tag{2.5}$$

where c_i represents the measurement bias associated to the anchor i. This bias may result from the measurement errors or disruptive environmental factors. It is usually used to correct distance measurements.

Multi-lateration localization methods are characterized by their high accuracy and increased resistance to interference. However, they can be sensitive to propagation delays and environmental changes, which can limit their precision under complex conditions.

(c) **Localization based on prediction using ML** : This technique uses ML algorithms to estimate the geographical position of an indoor object. It relies on the analysis of the data collected by the sensors to predict the position of an object in real time, which makes it an accurate and flexible localization method used in various application areas. This technique is applied following the steps presented below:

(i) Building a comprehensive dataset for training the model by collecting training data from objects, including measurements such as signal strength, time of flight, or environmental characteristics, as well as their known positions;

(ii) ML models, such as a DL model, decision tree and regression method, are generally trained on the training data to learn the relationship between data features and the corresponding localization positions. It is iteratively adjusted to minimize the error between the object predicted position and its actual one;

(iii) Validation of the model: is carried out through the evaluation of its performance on unseen data by examining validation data;

(iv) After being trained and validated, the model can be used to predict the localization position of a device or an object based on new real-time data. This data is fed into the model that, subsequently, estimates the position relying on the observed features.

ML-based location prediction is highly precise and flexible. However, it often requires a lot of training data and can be prone to errors in case of inadequate training of the prediction models or the change in the environmental conditions. Its implementation may also necessitates several and large computing resources to train the model.

(d) **Fingerprinting**: This method uses signal fingerprints to associate known locations with specific signal patterns, which allows determining the position of the object based on the measurements of the received signal. The application of the fingerprinting technique consists of the following steps:

(i) Collecting digital fingerprints involving the construction of a radio signal database from different points in the area to be localized;

(ii) A machine learning algorithm or signal processing method is utilized to create a model or a signal map from the data stored in the database;

(iii) To locate an object, the radio signals available at its current location should be measured. This signal data are, then, compared to the model or signal map created during the training phase. The location of the device is estimated by finding the closest match between the measured signal data and the signal model in the database.

The fingerprinting localization method is characterized by its high accuracy and adaptability to different environments. However, it requires an intensive learning phase to create generally large database and it can be affected by the variations in the radio- frequency environment.

(e) **Angular triangulation**: This technique is employed to estimate the position of an object based on the angles of the reception of signals sent by several known anchors distributed in the environment. Suppose that the environment contains n anchors with coordinates (x_i, y_i) for $i = 1, 2, ..., n$. The measurements of the angles θ_i between each anchor and the object to be localized must be taken (Figure 2.15) in order to estimate the position of the object.

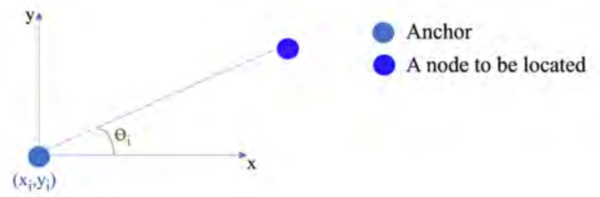


Figure 2.15: Measurement of the angle between an anchor i and the object to be located

To measure the angle of the arrival of the radio signal by evaluating the difference in the arrival time or the phase shift of the received radio modulation, the same receiver node should be equipped with two antennas. To achieve two-dimensional localization by angular triangulation, at least two anchors should be used. The functioning of this technique is demonstrated in Figure 2.16. The position of the object to be localized can be determined using the two angles θ_1 and θ_2 and by applying equation 2.6:

$$\begin{cases} \hat{x} = \frac{L\times\tan(\theta_2)}{\tan(\theta_2)-\tan(\theta_1)} \\ \hat{y} = \frac{L\times\tan(\theta_1)\times\tan(\theta_2)}{\tan(\theta_2)-\tan(\theta_1)} \end{cases} \tag{2.6}$$

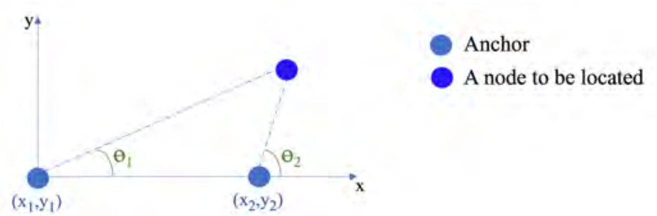


Figure 2.16: Angular triangulation with two anchors

Another localization scenario is possible: a node whose position is known and can measure a distance and an angle relative to a mobile object is, therefore, able to locate it.
Although angular triangulation localization is effectively used in many applications where precise angle measurements can be performed by providing high precision, it requires the use of precise angle measuring devices and can be sensitive to orientation or calibration errors. As connected objects are often very small, the distance between the two antennas of the same receiver is short, which prevents obtaining precise angle measurement.

2.4.2 Comparison and review of recent studies on indoor localization

In the literature, several studies focused on solving the indoor localization problem in IoT networks. Table 2.3 provides a comparison between recent approaches proposed to address this particular challenge.

2.4.3 Synthesis

Review of the afore-mentioned works showed that current methods generally failed to achieve the required location accuracy in IoT environments where sub-metric precision should be ensured in indoor location systems [42]. However, research works that achieved average errors below 1m obtained these results through simulations rather than real data and these simulations were not carried out using a simulator dedicated to IoT networks, taking into account specific radio links and their particular transmission technologies. On the other hand, in other works, the obtained accuracy did not meet the indoor localization requirements of IoT applications. In this context, we intend to propose an accurate indoor location solution that satisfies the requirements of IoT environments, based on real data.

2.5 Conclusion

In conclusion, the first chapter dealt with two key challenges of IoT networks: routing and indoor localization. We examined different aspects of routing, including its types, the used performance metrics and the recent studies on routing, as well as some key aspects of indoor localization (localization techniques, evaluation criteria and recent research works focusing on indoor localization). This chapter highlighted the importance of efficient routing and accurate localization in ensuring an optimal operation in IoT networks. In the next chapter, we will explore optimization and describe Deep Learning techniques. We will also demonstrate the process of applying these approaches to solve specific challenges encountered in IoT networks, particularly in terms of routing and indoor localization.

Table 2.3: Comparison and review of recent works solving the indoor localization problem in IoT networks

Paper	Year	Type	Approach/Objectives	Techniques	Technology	Measures	Results		Disadvantages
							MAE	Precision	
[43]	2023	range-based	predict positions inside a university building	DNN	LoRaWAN	RSSI, SNR	-	98.8%	The proposal was not compared to other solutions using RSSI and SNR measurements
[30]	2023	range-free	Develop CMWN-DVHop, an improved version of DV-HoP algorithm by introducing segmentation and weighting factors.	"Cyclotomic method" and Weighted Recursive Least-Squares (WRLS) algorithm	-	-	2.9m	-	No real-world experiments; Simulations are conducted using Matlab, rather than with a dedicated IoT network simulator.
[31]	2023	range-free	Propose WRCDV-Hop featuring four improvements to DV-HoP algorithm	Weighted method, Whale Optimization Algorithm (WOA)	-	-	0.13m	-	Results are based on simulations rather than real data; Simulations are conducted using Matlab, rather than a specific IoT network simulator.
[32]	2023	range-free	Propose a localization algorithm, HW-DV-HopCSO, an improved version of DV-HoP	Chicken Swarm Optimization (CSO)	-	-	-	80%	Accuracy obtained does not meet IoT application requirements; Results are obtained from simulations using Matlab, rather than a specific IoT network simulator.
[44]	2023	range-free	Optimize APIT localization algorithm by enhancing RSSI data preprocessing	Gaussian filtering, ANN	-	RSSI	1.55m	-	Average distance error obtained does not meet IoT application requirements; Results are from simulations conducted using Matlab instead of a dedicated IoT network simulator.
[45]	2022	range-based	Develop two indoor localization solutions	SVR and KF	-	RSSI	0.85m	-	No real-world experiments
[46]	2022	range-based	Develop a framework, CSILoc, including a DL solution for accurate localization estimation	Fingerprinting, CNN	Wi-Fi	CSI	1.75m	68.5%	Obtained results do not meet indoor IoT localization requirements.
[47]	2022	range-free	Propose an improved version of Triangle Centroid Localization algorithm	PIT	-	RSSI	1.42m	-	Results rely on simulations conducted using Matlab, rather than real data, and are not from a specific IoT network simulator; Average distance error obtained does not meet indoor IoT localization criteria.
[48]	2021	range-based	Propose a new ensemble model as an indoor localization solution	Fingerprinting, KNN, DNN, and RF	Wi-Fi	RSSI	1.10m	-	Average distance errors were not adequate to address the indoor localization challenge in IoT environments.

Continued on next page

Table 2.3 – continued from previous page

Paper	Year	Type	Approach/Objectives	Techniques	Technology	Measures	Results		Disadvantages
							MAE	Precision	
[49]	2021	range-based	Develop a 2D image serving as a basis for a DL model, then apply an improved PSO algorithm to optimize its training.	CNN and an improved PSO variant	BLE	RSSI	-	97.92%	Precision obtained is based on validation phase results. However, their reliability needs to be verified through a separate testing phase to confirm their validity.
[50]	2021	range-based	Develop an indoor localization solution (Poucet) tailored to the needs of firefighters and soldiers.	Trilateration	UWB, LoRa	ToF	0.36m	-	No real-world experiments
[51]	2021	range-based	Develop a UAV localization solution for applications in extremely confined environments.	MLE method	UWB	ToF	0.2m	-	Proposed solution performance is not evaluated against other existing solutions to demonstrate its effectiveness.
[52]	2021	range-based	Propose an indoor positioning system that can work on any smartphone with reduced temporal complexity	Fingerprinting, a proposed new Particle Filter algorithm	BLE	RSSI	1.4m	-	Efficiency of proposed solution is not compared to existing solutions to evaluate its performance.
[53]	2020	range-based	Propose a new indoor object localization and tracking solution	Fingerprinting, MLP	Wi-Fi	RSSI	-	83%	Results need experimental validation on data collected in real environment.
[54]	2020	Range-free	Develop and present a distributed, collaborative localization solution	A new UWL (Uncertainty Weighted Localization) algorithm	UWB	-	<1m	-	Essential to conduct practical experiments to confirm validity of these results.
[55]	2020	range-based	Propose an indoor localization solution called DELTA	Fingerprinting, DNN	5G	RSSI	1.6m	89%	Proposed model performance needs to be verified in real conditions and through an additional testing phase, not just through validation; Precision obtained does not meet IoT indoor localization requirements.

Chapter 3

Optimization and Deep Learning

Contents

3.1 Introduction

This chapter focuses on two areas that offer powerful and efficient tools used to solve several complex problems. It begins by defining the different types of optimization problems: single-objective and multi-objective problems. Then, it explores the main notions on which the presented solution methods are based. Afterwards, it describes the methods suggested to solve the optimization problems, particularly meta-heuristics techniques and hybrid methods. Subsequently, it presents the criteria used to evaluate them. The first section ends with a review about the recent methods applied to solve optimization problems. In the second part of this chapter, we are interested in DL. We depict the main areas of DL application, especially its use in regression. We also explore its theoretical foundations, including the operation of the training methods and that of parameter optimization techniques. Then, we present the main DL techniques that can be applied to deal with regression problems. Subsequently, the performance criteria used to evaluate these methods are discussed. The final part of this chapter highlights the recent advances and research trends in these domains (optimization and DL) as well as their use to solve the routing and localization problems in IoT networks.

3.2 Optimization problems

Optimization problems can be defined as the situations where it is necessary to select the best solution by maximizing or minimizing one or more objective functions among a set of possible solutions. The latter are generally defined by a search space and constraints. Optimization problems are encountered in many fields (e.g, engineering, economics, logistics, finance, etc.) where their effective resolution is crucial in decision making and process improvement. These problems can be classified into several categories according to the nature of the objective functions and the constraints. This first section presents classification based on the number of objectives.

3.2.1 Single-objective optimization problems

Single-objective optimization problems are problems where one objective function must be maximized or minimized under certain constraints. Mathematically speaking, a single-objective optimization problem can be formulated as follows:

$$\begin{aligned} \text{Minimize/Maximize} \quad & f(x) \\ \text{Subject to the constraints} \quad & g_i(x) \leq 0, \quad i = 1, 2, \ldots, m \\ & h_j(x) = 0, \quad j = 1, 2, \ldots, p \end{aligned} \tag{3.1}$$

where $f(x)$ is the objective function to be optimized, x denotes the vector of decision variables, $g_i(x)$ are inequality constraints, and $h_j(x))$ refer to equality constraints. The objective of its resolution is to find the optimal value of the vector x which minimizes or maximizes the objective function $f(x)$ while satisfying the constraints $g_i(x)$ and $h_j(x)$.

3.2.2 Multi-objective optimization problems

Multi-objective Optimization Problems (MOP) are problems in which several and often contradictory objective functions must be optimized simultaneously. In fact, a multi-objective optimization problem can be mathematically formulated as follows:

$$\begin{aligned} \text{Minimize/Maximize} \quad & f_i(x), \quad i = 1, 2, \ldots, m \\ \text{Subject to the constraints} \quad & g_j(x) \leq 0, \quad j = 1, 2, \ldots, n \\ & h_k(x) = 0, \quad k = 1, 2, \ldots, p \end{aligned} \tag{3.2}$$

where $f_i(x)$ are the objective functions to be optimized, x is the vector of decision variables, $g_j(x)$ represent inequality constraints, and $h_k(x)$ designate equality constraints. The objective of its resolution is to find the decision vector x which simultaneously minimizes or maximizes all objective functions $f_i(x)$ while satisfying the constraints $g_j(x)$ and $h_k(x)$.
In the remainder of this section, we consider an optimization problem as a minimization problem.

3.2.3 Main used principles

Let S be the set of feasible solutions and $F(X)$ be the set of objective functions to be optimized in a multi-objective optimization problem with m objectives. A feasible solution is one that is within the feasible region of the problem and meets all of its constraints. It does not have to be optimal, but it must adhere to the rules set by the constraints.

3.2.3.1 Dominance

The notion of dominance is used in multi-objective optimization problems to compare the different possible solutions. A solution x_1 is said to be dominant compared to another solution x_2 if it offers values that are better than or equal to x_2 for all objectives and strictly better than x_2 in, at least, one objective. Formally, let $f(x_1) = (f_1(x_1), f_2(x_1), ..., f_m(x_1))$ and $f(x_2) = (f_1(x_2), f_2(x_2), ..., f_m(x_2))$ be the objective vectors associated to the solutions x_1 and x_2, respectively, and x_1 dominates x_2 if and only if:

$$\begin{aligned}
&f_i(x_1) \leq f_i(x_2) \quad \text{for each} \quad i \in \{1, 2, ..., m\} \\
&\text{and} \\
&\exists\, i \text{ such that} \quad f_i(x_1) < f_i(x_2)
\end{aligned} \tag{3.3}$$

This dominance is expressed by: $x_1 \succ x_2$. Figure 3.1 illustrates an example of the concept of dominance where the star is dominated by the circles; the star dominates the squares and the star is equivalent to triangles.

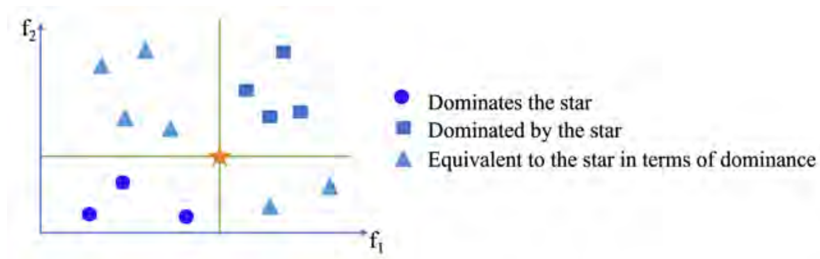


Figure 3.1: An example of the concept of dominance

To take into account constraints in an optimization problem, the notion of dominance must be adapted. A solution x_1 dominates another x_2 if one of the following three situations is true:

- x_1 and x_2 satisfy all constraints and x_1 dominates x_2. Formally:

$$\begin{cases}
g_j(x_1) \leq 0 \quad \text{for each} \quad j = 1, 2, \ldots, n \quad \text{and} \quad h_k(x_1) = 0 \quad \text{for each} \quad k = 1, 2, \ldots, p \\
g_j(x_2) \leq 0 \quad \text{for each} \quad j = 1, 2, \ldots, n \quad \text{and} \quad h_k(x_2) = 0 \quad \text{for each} \quad k = 1, 2, \ldots, p \\
x_1 \succ x_2
\end{cases} \tag{3.4}$$

- x_1 satisfies all the constraints while x_2 does not. Formally:

$$\begin{cases} g_j(x_1) \leq 0 \quad \text{for each} \quad j = 1, 2, \ldots, n \quad \text{and} \quad h_k(x_1) = 0 \quad \text{for each} \quad k = 1, 2, \ldots, p \\ \exists\, j \text{ such that} \quad g_j(x_2) > 0 \quad \text{or} \quad \exists\, k \text{ such that} \quad h_k(x_2) \neq 0 \end{cases} \tag{3.5}$$

- x_1 and x_2 do not satisfy all constraints, but x_1 presents fewer constraint violations than x_2. Formally:

$$\begin{cases} \exists\, u_1 \text{ such that} \quad u_1 \subset \{1, 2, \ldots, n\} \quad \text{and} \quad g_j(x_1) > 0 \quad \text{for each} \quad j \in u_1 \\ \text{and/or} \quad \exists\, v_1 \text{ such that} \quad v_1 \subset \{1, 2, \ldots, p\} \quad \text{and} \quad h_k(x_1) \neq 0 \quad \text{for each} \quad k \in v_1 \\ \text{and} \\ \exists\, u_2 \text{ such that} \quad u_2 \subset \{1, 2, \ldots, n\} \quad \text{and} \quad g_j(x_2) > 0 \quad \text{for each} \quad j \in u_2 \\ \text{and/or} \quad \exists\, v_2 \text{ such that} \quad v_2 \subset \{1, 2, \ldots, p\} \quad \text{and} \quad h_k(x_2) \neq 0 \quad \text{for each} \quad k \in v_2 \\ \text{and} \\ |u_1| + |v_1| < |u_2| + |v_2| \end{cases} \tag{3.6}$$

The notion of dominance allows determining which solutions are Pareto optima.

3.2.3.2 Pareto Optimum

Pareto optimum is a concept used in multi-objective optimization to define solutions that cannot be improved in all objectives without deteriorating, at least, one other objective. Formally, a solution x^* is considered as a Pareto optimum if and only if, for all x belonging to S, there is no other point y in S such that $y \succ x$. The set of all Pareto solutions is called the Pareto front.

3.2.3.3 Pareto front

Pareto front represents all potentially-optimal solutions which offer a compromise between the different objectives to be optimized. In other words, no point in the Pareto front can be enhanced in all objectives without degrading, at least, one objective. The Pareto front is defined below:

$$PF_{true} = \{x \in S \mid \nexists\, x' \in S, \forall f \in F(X), f(x') \leq f(x)\} \tag{3.7}$$

Figure 3.2 shows examples of Pareto fronts for two- and three-objective problems.

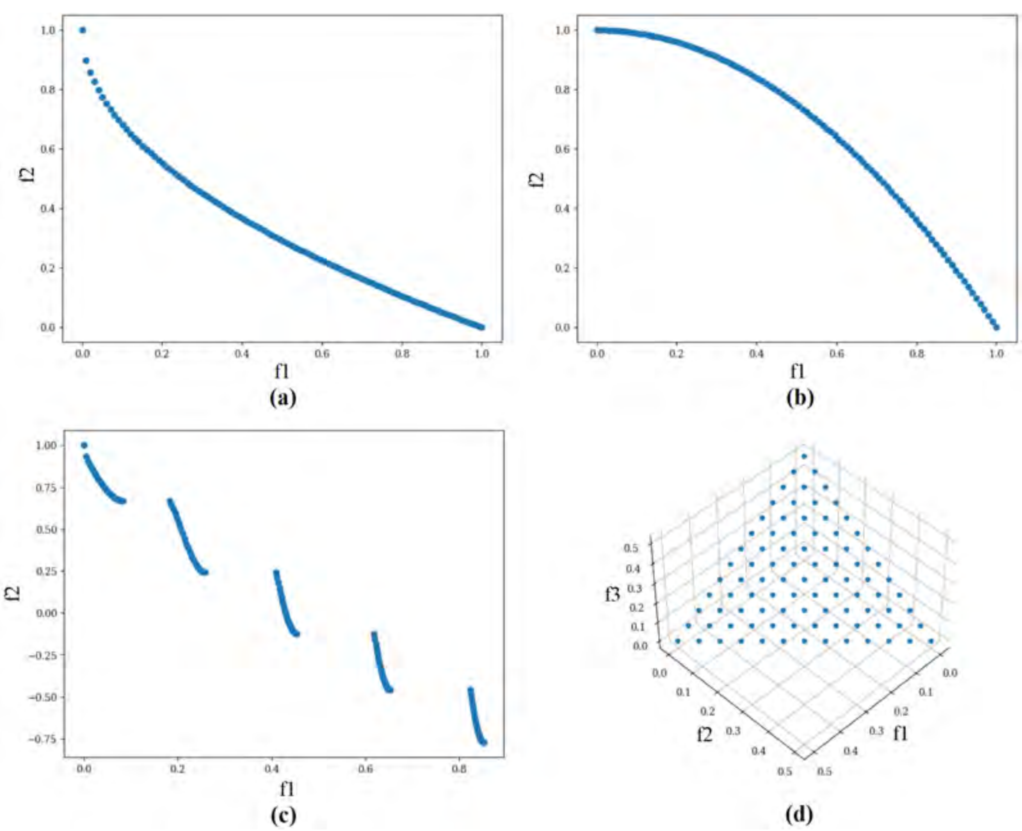


Figure 3.2: Examples of the different geometric shapes of Pareto fronts. For 2 objectives : (a) convex ; (b) concave ; (c) disjoint, for 3 objectives : (d) linear

3.2.3.4 The ideal point

The ideal point is located in the space of objectives which represents the best possible value of each objective. It is often used as a reference to evaluate the performance of the solutions of a multi-objective problem.
The ideal point is mathematically, defined as a vector $z^* = (z_1^*, z_2^*, ..., z_m^*)$ such that:

$$z_k^* = \min\{f_k(x) \,|\, x \in S\}, \quad \text{pour tout } k = 1, 2, ..., m \tag{3.8}$$

Figure 3.3 illustrates the ideal point which does not belong to the space of the feasible solutions because the objectives are generally contradictory.

3.2.3.5 The anti-ideal point

The anti-ideal point can be defined as the maximum values of each objective functions taken separately for all the possible solutions in the search space (Figure 3.3). It is represented as follows:

$$z^a = (\max f_1(x), \max f_2(x), \ldots, \max f_m(x)) \tag{3.9}$$

where $f_i(x)$ denotes the i-th objective function, and x is a possible solution in the search space.

3.2.3.6 The Nadir point

The Nadir point $N = (N_1, N_2, ..., N_m)$ is defined as the vector of the upper bounds of each objective on the Pareto front (Figure 3.3) such that:

$$N_i = \max_{x \in PF} f_i(x) \quad \text{for each} \quad i = 1, 2, ..., m \tag{3.10}$$

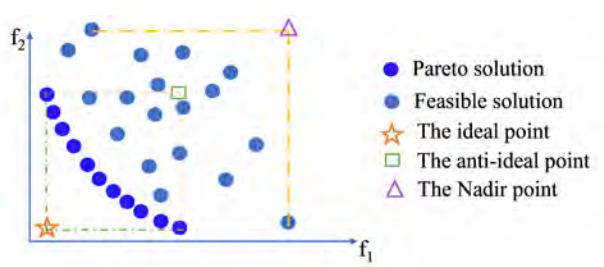


Figure 3.3: Illustration of the ideal point, anti-ideal point, and Nadir point

3.2.4 Resolution methods

Several methods and techniques can be applied to solve multi-objective optimization problems by finding a set of solutions that represent the optimal compromise between the different objectives.

3.2.4.1 Types of resolution methods

Many types of solution methods were proposed in the literature to solve multi-objective optimization problems. The most common techniques are:

- The exact methods: are approaches employed to find all the optimal solutions of the problem using formal mathematical techniques. Despite their efficiency, they often suffer from computational complexity and limitations in terms of the problem size. Although they guarantee optimal solutions, they are inefficiently utilized to solve large problems due to their exponential complexity.
  - Exhaustive enumeration methods, such as brute-force method and the exhaustive decision tree search method, consist in enumerating all the possible solutions of the MOP and evaluating them to find the optimal solution.
  - Linear Programming (LP) techniques: They are based on formulating the MOP in the form of a linear program with multiple objectives and linear constraints. Then, algorithms, such as the Simplex method, can be used to solve the problem. For instance, the Extended Multi-objective Simplex Algorithm (EMSA) [56] combines the principles of Simplex, used to solve the primal problem, and Duality, applied to determine the optimization directions of each objective, in order to find the front of Pareto optimal to solve Linear Multi-objective Optimization Problem (LMOP).
  - Mixed-Integer Quadratic Programming (MIQP): When the MOP includes quadratic terms in the objective function or in the constraints, it can be formulated as a MIQP problem. MIQP techniques (e.g., decomposition-based solution methods) can be used to solve such problem. The Benders Decomposition method [57] is an example of these methods.
- Weighting methods: are techniques employed to solve multi-objective optimization problems by transforming them into single-objective optimization problems. They consist in assigning weights to the different objectives to aggregate them into one weighted objective function. The weights are generally selected to reflect the relative importance of the objectives. After defining this weighted function, standard single-objective optimization methods will be applied to find a solution. Despite the simple application of the weighting techniques, the latter

are sensitive to the choice of weights and do not always guarantee efficient solutions on the Pareto front.

- Metaheuristics: are approaches that rely on empirical rules and approximation techniques to find the potential solutions without guaranteeing optimality. Unlike the exact methods, these techniques do not ensure the exhaustive search of all possible solutions. However, they often offer an efficient alternative to quickly explore a large search space and find an effective solution.

- Hybrid methods: combine the advantages of different resolution approaches, especially exact techniques or ML techniques with metaheuristics, to solve optimization problems more efficiently by exploiting the complementarity of the various applied approaches.

In the remainder of this section, we will define and present metaheuristic and hybrid methods.

3.2.4.2 Metaheuristics

Metaheuristics are approximate methods used to find solutions close to optimality without guaranteeing convergence towards an optimal global solution. These approaches are often employed to solve complex optimization problems and when the convergence of the exact methods is time consuming. Approximate methods are based on various algorithms; the most important of which are listed in this section.

(a) Bio-inspired metaheuristics : Bio-inspired metaheuristics are inspired from different behaviors observed in nature to solve optimization problems.

- **An example of bio-inspired metaheuristics: the GWO**

  GWO [3] is a bio-inspired algorithm recently introduced to solve single-objective optimization problems. It is inspired from the social behavior of gray wolves during group hunting. Algorithm 1 describes how it operates during resolution. In fact, it was designed to find the optimal solution for an optimization problem by iteratively updating the positions of the wolves representing the solutions which constitute the search space:

  - Wolves α, β and δ represent the optimal solutions since they have better knowledge about the location of the prey as they guide the hunting process and dominate the ω wolves;
  - Wolf α is the best solution;
  - Wolves ω represent the other solutions.

- **An improved version of GWO: Improved Gray Wolf Optimizer (I-GWO)**

  An improved version of GWO, called I-GWO, was proposed in 2021 in [58]. In I-GWO, two candidate positions are constructed to calculate and select $X_i(t+1)$. They are described below:

Algorithm 1: Gray Wolf Optimizer

Result: Best found solution
Initialization: Generate a population of gray wolves with random positions
Initialize coefficients: a, A, and C
Evaluate the fitness of each wolf X_i in the population
Identify the wolves X_α, X_β, and X_δ based on their fitness values
while *termination criterion not met* **do**
  for *each wolf in the population* **do**
    Update wolf's position:
    $\overrightarrow{D_\alpha} = |\overrightarrow{C_1}.\overrightarrow{X_\alpha} - \overrightarrow{X_i}(t)|, \overrightarrow{D_\beta} = |\overrightarrow{C_2}.\overrightarrow{X_\beta} - \overrightarrow{X_i}(t)|, \overrightarrow{D_\delta} = |\overrightarrow{C_3}.\overrightarrow{X_\delta} - \overrightarrow{X_i}(t)|$
    $\overrightarrow{X_{i1}} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.(\overrightarrow{D_\alpha}), \overrightarrow{X_{i2}} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.(\overrightarrow{D_\beta}), \overrightarrow{X_{i3}} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.(\overrightarrow{D_\delta})$
    $\overrightarrow{X_i}(t+1) = (\overrightarrow{X_{i1}} + \overrightarrow{X_{i2}} + \overrightarrow{X_{i3}})/3$
  Update coefficients
  Evaluate the fitness of each wolf in the population
  Identify the wolves X_α, X_β, and X_δ based on the new fitness values
return X_α

- The first candidate position, called $X_{i-GWO}(t+1)$, is computed using the same calculation method used in GWO.
- The second candidate position, named $X_{i-DLH}(t+1)$, is calculated employing a search strategy called Dimension learning-based hunting (DLH). This position is computed from the coordinates of the neighbors of $X_i(t)$ and a wolf chosen randomly from the elements of the search space.

Then, one of the two candidates is selected. The position chosen from $X_{i-GWO}(t+1)$ and $X_{i-DLH}(t+1)$ represents $X_i(t+1)$. X_i is only modified if the fitness value of the new position is lower than that of the current position.

- **Multi-objective version of GWO: MOGWO**

A multi-objective version of GWO, called MOGWO, was developed in 2016 in [2]. Two notions were added to GWO in order to deal with multi-objective optimization:

(i) The first notion is the use of an archive which is a space where non-dominated solutions are stored. It has a maximum capacity to contain all solutions. The archive consists of two main elements:
  - A controller whose role is to control the addition of a new non-dominated solution, NS, to the archive:
    * At first, the archive is empty;
    * If the archive is empty, the new solution NS can be accepted;
    * If there is a solution in the archive that dominates NS, the latter will be rejected;
    * If there is no solution in the archive that dominates NS, the latter will be accepted;

* If NS and the elements of the archive do not dominate each other, the new solution will be accepted;
* If there are items in the archive that are dominated by the newly-added solution, they should be removed.

- A grid :
  Solutions in the archive are distributed and placed in an m-dimensional objective functions space (recall that m is the number of objective functions of the problem to be resolved). The position of a solution in the objective functions space is determined according to its objective values.
  An m-dimensional grid is utilized to track the degree of crowding in different regions of the objective functions space. In fact, the grid is subdivided into a pre-determined number of hypercubes (segments). When a solution is added to the archive, its position in the grid and, therefore, the hypercube to which it belongs are calculated. The roles of the grid consist in distributing properly the solutions in the space of the objective functions and maintaining their diversification. Indeed, if a new solution is added, it is forwarded to the part of the grid containing the smallest number of elements. When the archive becomes full, a new solution can be added only after deleting an element from the archive. In this case, the grid mechanism is triggered to re-arrange the space segmentation and remove a solution from the most crowded segment. Afterwards, the new solution is inserted into the least crowded segment.

(ii) The second notion is a mechanism applied to select the best solutions. In fact, in the GWO algorithm, the 3 best solutions X_α, X_β and X_δ are chosen. Then, they will guide the agents during the search for the optimum. On the other hand, in a multi-objective problem, solutions cannot be compared in a simple way. To overcome this weakness, MOGWO provides a mechanism that allows selecting solutions X_α, X_β and X_δ from the elements of the least crowded segments. This choice makes it possible to direct the search for solutions towards unexplored areas. The selection method consists of:

- Selecting the least-crowded segment;
- If the number of solutions in the selected segment is $\geq$ to 3, three solutions will be chosen randomly and assigned to X_α, X_β and X_δ;
- If the number of solutions in the selected segment is $<$ to 3, the other solutions will be selected from the elements of the second least-crowded segment;
- If the second least-crowded segment contains only one solution, the third least- crowded segment must be found to assign one of its elements to X_δ.

To sum up, bio-inspired metaheuristics are flexible and robust approaches applied to solve several optimization problems. Although they provide efficient search in large solution spaces, their performance is affected by the type of the problem to be resolved and often requires careful adjustment of their parameters. Although they do not always guarantee convergence towards the optimal solution, these methods remain widely used thanks to their ability to find good solutions within reasonable time frames.

(b) **Genetic algorithms** : Genetic algorithms are optimization techniques inspired by the process of natural selection and genetics. The resolution of an optimization problem using genetic algorithms is based on the following major steps (Figure 3.4):

- **Population initialization**: an initial population of candidate solutions is randomly generated. Each solution x_i is represented as a sequence of genes or values;
- **Fitness evaluation**: Each solution in the population is evaluated relying on its fitness with respect to the objectives of the optimization problem. This assessment is carried out by applying a fitness function $f(x_i)$ which measures the performance of each solution;
- **Parent selection**: The most suitable solutions are selected to become the parents of the next generation using several selection methods such as tournament selection, roulette selection or rank selection;
- **Reproduction**: The selected parents reproduce to form a new generation of solutions through genetic operations such as recombination (or crossing over) and mutation. Crossbreeding combines genetic information of parents to create offspring, while mutation introduces small random variations in chromosomes to maintain genetic diversity;
- **Population assessment**: The new population, made up of descendants and sometimes some members of the previous generation, is assessed based on their fitness;
- **Parent replacement**: The worst performing members of the population can be omitted and replaced by new offspring employing many replacement strategies such as generational replacement or survival replacement;
- **Stopping criterion**: The evolution process continues during a certain number of iterations or until reaching a pre-defined stopping criterion. The latter is based on the number of iterations, the convergence of the population towards an optimum or other considerations related to the problem to be resolved.

For example, the Non-dominated Sorting Genetic Algorithm III (NSGA-III), proposed in [59], is a multi-objective genetic algorithm that utilizes an improved ranking method to rank solutions according to their performance on multiple objectives. Its main principles are described in 2. The

Figure 3.4: The major steps of a genetic algorithm

algorithm begins with initializing a population of random solutions. Then, these solutions are selected, crossed and mutated to form a new intermediate population. The objectives of each solution in this intermediate population are evaluated. Then, the NSGA-III ranking method is used by taking into account several objectives and similarity between solutions. This method involves a non-dominance step where non-dominated solutions are ranked in the first Pareto front, while dominated solutions are rejected. For each Pareto front, the population density is calculated employing the Crowding distance. Subsequently, the solutions are sorted based on the population density. The sorted solutions are, then, concatenated to form a ranked population. Finally, a niche selection method is applied to choose a diverse and well-distributed final population. This technique involves dividing the search space into niches and, afterwards, selecting the solution closest to the Pareto front in each niche. If the population size is smaller than desired, additional solutions will be randomly selected in each niche. The final population is, then, sorted using the criteria of non-dominance and population density again.

Algorithm 2: Non-dominated Sorting Genetic Algorithm III

Result: A population of non-dominated solutions
Initialization: Create an initial population P_0 of size N of random solutions
while *termination criterion not met* **do**
  Selection: Select N solutions from P_{t-1} to form a temporary population R_t
  Crossover: Apply the crossover operator to R_t to generate an intermediate population Q_t
  Mutation: Apply the mutation operator to Q_t to generate an intermediate population Q'_t
  Evaluation: Evaluate the objectives of all solutions in Q'_t
  Ranking: Rank the solutions in $P_{t-1} \cup Q'_t$ using the NSGA-III ranking method
  Niche: Select N solutions from $P_{t-1} \cup Q'_t$ using the niche selection method to form population P_t
return P_t

Genetic methods ensure efficient exploration of the search space and can handle multiple solutions simultaneously. Although they are simple to implement, they converge slowly. They are also sensitive to chosen values of their parameters and exhibit reduced performance when applied to solve large-scale problems.

(c) Particle Swarms Particle Swarm methods are optimization algorithms based on models inspired from the social behavior of animals in groups, such as swarms of birds or schools of fish, and their self-organized interactions. They are utilized to find the best solution of a given problem by iterating over a set of particles; each of which represents a potential solution. Each particle is characterized by a position in the search space and an associated velocity.

- **Particle Swarm Optimization (PSO)** [60] is an example of these algorithms used to solve single-objective optimization problems. It is population-based algorithm inspired from the social behavior of bird flight. Algorithm 3 represents the principle of PSO. Its objective consists in solving a problem by iterating over a set of particles, each of which represents a potential solution having a position and a velocity in the search space. The positions and velocities of the particles are initially random. In each iteration, each particle updates its position and velocity according to its best position and the best overall position among all particles by employing inertia and acceleration factors as well as random values. The algorithm continues until a stopping criterion, such as the maximum number of iterations, is reached. Then, it returns the best overall found position representing the optimal solution. In this algorithm, $x_{i,d}(t)$ represents the position of the particle i in dimension d at iteration t, and $v_{i,d}(t)$ is the corresponding speed. The own best position of the particle i in dimension d is denoted by $p_{i,d}$, and $p_{g,d}$ corresponds to the global best position among all particles in dimension d. ω is the inertia weight, while c_1 and c_2 are acceleration constants, and r_1 and r_2 are random values uniformly distributed in [0, 1].

Algorithm 3: Particle Swarm Optimization

Result: Optimal solution
Initialization: Initialize particles P_0 with random positions and velocities
while *termination criteria not met* **do**
  for *each particle i* **do**
    Update personal best position: $p_i \leftarrow$ best position of particle i
    Update global best position: $p_g \leftarrow$ best position among all particles
    for *each dimension d* **do**
      Update particle velocity:
      $v_{i,d}(t+1) \leftarrow \omega \cdot v_{i,d}(t) + c_1 \cdot r_1 \cdot (p_{i,d} - x_{i,d}(t)) + c_2 \cdot r_2 \cdot (p_{g,d} - x_{i,d}(t))$
      Update particle position: $x_{i,d}(t+1) \leftarrow x_{i,d}(t) + v_{i,d}(t+1)$
    Evaluate the new position
return *Global best position p_g*

The PSO algorithm is characterized by its simplicity, ease of implementation and rapid convergence. It has been applied to solve various optimization problems including continuous, discrete and combinatorial optimization. However, it suffers from feedforward convergence and stagnancy, especially when applied to deal with high-dimensional optimization problems.

To solve these problems, various modifications and extensions of the PSO algorithm were proposed. Some of them are described below:

- The Bare Bones Particle Swarm Optimization (BBPSO) [61]: is a modified version of PSO employed to simplify the equations and reduce the number of parameters. Its goal is to find a balance between exploration and exploitation by dynamically adjusting the searching behavior of particles.
- The Quantum-behaved Particle Swarm Optimization (QPSO) [62]: is a variant of PSO that uses an approach inspired from quantum physics to update the positions and velocities of particles, which leads to better convergence and more robustness.

• **Multi-objective Particle Swarm Optimization (MOPSO)**: MOPSO [60] is a new version of PSO designed to handle multi-objective optimization problems. Indeed, it introduces a representation of solutions taking into account several objectives simultaneously, uses a domination function to evaluate the efficiency of the solutions according to these objectives, maintains an archive of non-dominated solutions to form the Pareto front and updates the best overall position by considering this archive. These additions allow MOPSO to efficiently solve multi-objective problems by exploring a diverse set of optimal solutions. Algorithm 4 represents the different steps of MOPSO.

  Dual Multi-Objective Particle Swarm Optimization (DMOPSO) is an improved version of MOPSO developed by [63]. It includes additional goals to optimize both user's preference and diversity.

PSO methods are a simple and accessible approached applied to deal with various optimization problems thanks to their easy implementation and their effectiveness in exploring large search spaces. They can quickly converge to best solutions. However, they can also be parameter sensitive and tend to converge to local optimal solutions in complex search spaces.

(d) **Evolutionary programming (EP)**: Evolutionary programming algorithms mimic the process natural selection and reproduction within a population of candidate solutions. For example, we can cite the Electrostatic Potential Energy Evolutionary Algorithm (ESPEA) [64] and the Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [65]. ESPEA uses the concept of electrostatic potential energy to estimate the solutions obtained at (PF_{true}). On the other hand, MOEA/D is widely utilized to solve multi-objective optimization problems. It divides a multi-objective problem into several single-objective sub-problems and optimizes them simultaneously. This approach consists in creating a set of weight vectors, that uniformly cover the objective space, and optimizing the sub-problems associated to these weight vectors. The solutions of these sub-problems are, then, combined to form a varied and well-converged population that approximately represents the Pareto front.

Algorithm 4: Multi-objective Particle Swarm Optimization

Result: Set of non-dominated solutions P
Initialization: Generate a population of N particles with random positions $\mathbf{x}_i$ and velocities $\mathbf{v}_i$
for $i = 1$ *to* N **do**
  Evaluate the performance of particle i using objective functions $f_m(\mathbf{x}_i)$, where $m = 1, 2, \ldots, M$
  if $f_m(\mathbf{x}_i) < f_m(\mathbf{p}_i)$ *for all* m **then**
    $\mathbf{p}_i = \mathbf{x}_i$; $F(\mathbf{p}_i) = F(\mathbf{x}_i)$
Update global best positions $\mathbf{g}_m$ and performance $F(\mathbf{g}_m)$ of the swarm based on the performance values of all particles
while *termination criterion not met* **do**
  for $i = 1$ *to* N **do**
    Update the velocity and position of particle i using the equations:
    $\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1 r_1(\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2(\mathbf{g}_m(t) - \mathbf{x}_i(t))$
    $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1)$
    Evaluate the performance of the new position of particle i using objective functions $f_m(\mathbf{x}_i)$, where $m = 1, 2, \ldots, M$
    if $f_m(\mathbf{x}_i) < f_m(\mathbf{p}_i)$ *for all* m **then**
      $\mathbf{p}_i = \mathbf{x}_i$
      $F(\mathbf{p}_i) = F(\mathbf{x}_i)$
    if $f_m(\mathbf{x}_i) < f_m(\mathbf{g}_m)$ *for all* m **then**
      $\mathbf{g}_m = \mathbf{x}_i$
      $F(\mathbf{g}_m) = F(\mathbf{x}_i)$
  Update the set of non-dominated solutions P using the following rules:
  if *particle i is not dominated by any other particle in P* **then**
    Add particle i to P
  else if *particle i is dominated by another particle in P* **then**
    Remove particle i from P
  else if *particle i is in P and is dominated by another particle in P* **then**
    Remove particle i from P
return P

The Multi-Objective Evolutionary Algorithm based on Decomposition and Dominance (MOEA/DD) [66] is an improved version of MOEA/D. Algorithm 5 describes the resolution of a MOP by applying MOEA/DD. It initializes the parent population P, the set of weight vectors W and the set of neighborhood indexes E. Then, if the stopping criterion is not satisfied, it selects the breeding parents from the entire population with low probability $1 - \lambda$ where $\lambda \in [0, 1]$. Subsequently, it applies the simulated binary crossover (SBX) and the polynomial mutation to obtain a set S of several solutions. For each solution in S, the algorithm updates the parent population P. Finally, it returns the updated population P.

Many modified variants of MOEA/D, including Improved Epsilon Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D-IEps) [69], Epsilon Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D-Eps) [70], MOEA/D-SR [71] and MOEA/DCDP [71], were developed in previous research works.

Algorithm 5: Multi-Objective Evolutionary Algorithm based on Decomposition and Dominance

Result: Population P

Initialization:

- Initialize P: the parent population
- Initialize W: the set of weight vectors
- Initialize E: the set of neighborhood indexes

while *termination criterion is not met* **do**
    for $i = 1, 2, \ldots, N$ **do**
        Reproduction selection: select parent reproducers from the entire population with a low probability $1 - \lambda$, where $\lambda \in [0, 1]$
        Variation: apply Simulated Binary Crossover (Simulated Binary Crossover) [67] and polynomial mutation [68] to obtain S
        foreach $x \in S$ **do**
            $P \leftarrow$ update_population(P, x)
return P

Obviously, evolutionary programming can be considered as a robust and flexible approach employed to solve numerous optimization problems. It allows finding efficient solutions to deal with complex and dynamic problems as well as multi-objective problems and nonlinear constraints. EP is less parameter-sensitive and more extensively used, compared to other optimization techniques. However, it suffers from some weaknesses such as slow convergence, the need for intensive evaluations of the objective function, and difficulty in setting the parameters.

(e) Simulated Annealing : Simulated Annealing algorithms are stochastic optimization algorithms inspired from the process of cooling and crystallizing a molten material to reach a minimum energy state, which is similar to finding the optimal solution in a search space. The algorithm starts with an initial solution and explores the solution space by accepting or rejecting the neighboring solutions according to a probability determined by a temperature function. The temperature decreases over time, reducing the probability of accepting solutions worse than the current one, which makes the algorithm able to converge into an optimal or almost optimal solution. Simulated Annealing (SA) [72] is inspired from these algorithms. It is designed to solve single-objective problems. Algorithm 6 describes its principle. It starts by a random initial solution S and an initial temperature T. In each iteration, the algorithm generates a neighboring solution S' and evaluates its efficiency, in comparison to that of the current solution. The decision to accept or reject the neighboring solution depends on the cost difference ΔE and the current temperature T. Subsequently, temperature is updated according to a cooling pattern at each iteration. The algorithm stops once the temperature reaches its predefined minimum value T_{min}, and it returns the best found solution.

[72] developed an improved version of SA called Orikaeshi Tanren Simulated Annealing (OTSA). The suggested algorithm uses folding and heating to design two new operators. The folding op-

Algorithm 6: Simulated Annealing

Result: Best found solution
Initialization: Generate a random initial solution S
Initialize the temperature T to an initial value
while $T > T_{min}$ **do**
    Generate a neighboring solution S' from S
    Calculate the cost difference $\Delta E = E(S') - E(S)$
    if $\Delta E < 0$ **then**
        Accept the neighboring solution S'
    else
        Generate a random number p between 0 and 1
        Calculate the acceptance probability $P = e^{-\Delta E/T}$
        if $p < P$ **then**
            Accept the neighboring solution S'
        else
            Reject the neighboring solution S'
    Update the temperature T according to the cooling schedule
return S

erator gradually reduces the size of the search space, which increases the exploitation by the concentration of the search agents. At the same time, the heating process restarts cooling once the stopping criterion is reached, allowing the search agents to move more in order to avoid local optimal solutions.

Simulated annealing algorithms are simple and efficient approach applied to solve complex problems by avoiding local optimal solutions using the decreasing temperature strategy. They do not require prior knowledge about the problem structure and can provide efficient solutions. However, their performance can be affected by adjusting their parameters as well as by the dimensionality and complexity of the problem, without guaranteeing convergence towards the optimal solution.

(f) The Tabu Search (TS) : The Tabu Search method [73] is an approximate (metaheuristic) local search technique applied to solve MOPs. It is based on exploring the search space by generating new solutions from the existing ones. In this technique, the solutions that have already been visited are stored in a tabu memory list used to prevent the search from returning to the already-visited solutions. Algorithm 7 represents the steps followed to solve a MOP by applying the tabu search method. It starts by initializing a solution (s^*) randomly and creates an initially-empty tabu memory (POP). The Evaluate() function assesses the fitness value of the initial solution. Afterwrards, the algorithm maintains an optimal solution (S) and an iteration counter (it). At each iteration, it searches a set of neighboring solutions applying the findNeighborSet() function, finds the best solution in this set using the findBest() function, updates both the tabu memory by the update() function and the optimal solution S. This process repeats until the maximum number of iterations, defined by max_it, is reached. The algorithm aims at exploring the space of solutions while avoiding prohibited movements stored in the tabu memory to converge towards an optimal solution.

Algorithm 7: Tabu search algorithm for multi-objective optimization problems

```
Result: Optimal solution S
s* ← InitializeSolution()
Initialize tabu memory
POP ← null
Evaluate(s*)
S ← s*
it ← 0
while it ≤ max_it do
    it ← it + 1
    nbs ← FindNeighborhoodSet(s*)
    b ← FindBest(nbs, it)
    update(POP, it)
    S ← b
```

The tabu search ensures efficient exploration of the space of solutions, escaping local optimal solutions and adapting to complex problems. However, it can be sensitive to parameters, complex to implement and dependent on initial conditions.

3.2.4.3 Hybrid methods

Hybrid methods often combine different metaheuristics or exact techniques and ML techniques with metaheuristics to better solve optimization problems by exploiting the complementary advantages of the different combined approaches. This subsection presents two examples of hybridization approaches.

- **Example 1: genetic algorithm + tabu search**: [74] proposed a hybrid algorithm that combines a genetic algorithm with the tabu search to solve the Flow Shop Scheduling Problem (FSSP). The main objective of this combination is to minimize the makespan, i.e. the total time required to accomplish all tasks in the workshop. The flowchart representing the hybrid algorithm is exposed in Figure 3.5. It starts by generating an initial population of solutions via the genetic algorithm. Subsequently, it scans the population and applies the tabu search algorithm to each solution. If an improved solution is found, it will be integrated into the population and the tabu list will be updated. The algorithm stops as a solution with a fitness value lower than a predefined threshold is detected. The authors also provided experimental findings to evaluate the effectiveness of the developed hybrid algorithm. They compared the performance of this algorithm with that of other methods such as a genetic algorithm, the tabu search algorithm and a local search algorithm. The provided results prove that the hybrid algorithm outperforms other techniques in terms of finding the optimal solution and minimizing the makespan, which highlights that better performance can be obtained by combining two techniques. However, the introduced algorithm suffers from some limitations, notably its use of two metaheuristic algorithms, which prevents finding always the optimal solution, especially for complex problems. Moreover, its employment is restricted to solve the flow shop problem. It is also computationally expensive and difficult to parallelize.

Figure 3.5: The flowchart of the hybrid algorithm: a genetic algorithm + tabu search

- **Example 2, particle swarms + Machine Learning**: [75] suggested a hybrid algorithm to solve the complex problem of planning and resource allocation in fog computing environments. The Enhanced Multi-Objective Particle Swarm Optimization with Clustering (EMOPSOC) algorithm combines MOPSO with Machine Learning techniques to optimize fog node scheduling by minimizing latency between IoT devices and fog nodes, reducing waste of resources, balancing workloads, and improving the efficient use of resources.

  Figure 3.6 presents the EMOPSOC organization chart. The algorithm begins by initializing a population of particles with random positions and velocities. Then, it evaluates the fitness for each particle. Afterwards, the best personal positions and their fitness values are updated. The ML clustering technique is, subsequently, applied to group the particles into clusters as a function of their similarities. In each cluster, the best overall particle is selected for each particle. The velocities and positions of each particle are updated utilizing the best personal and global positions. The process continues until a stopping criterion, such as a maximum number of iterations or convergence, is reached. The performed simulations show a significant improvement, in the performance of the proposed algorithm in terms of finding a set of non-dominated solutions, maximizing the total efficiency of the problem of planning the resources and the placement of the applications and minimizing its total cost. However, EMOPSOC has some disadvantages including increased computational complexity due to the clustering step, difficulty in choosing the appropriate clustering method, risk of premature convergence to suboptimal solutions and limited exploration of the search space, which makes it difficult to predict its performance on different problems.

Figure 3.6: The flowchart of the hybrid EMOPSOC algorithm : MOPSO + ML clustering technique

Combining metaheuristics with exact methods and ML techniques is beneficial in the sense that it ensures improved solution accuracy, more efficient exploration and exploitation of the search space, better scalability, and increased flexibility and adaptability to changing requirements. However, this approach has some disadvantages including increased algorithm complexity, higher computational cost. It also faces challenges in choosing the appropriate combined methods.

3.2.5 Criteria used to evaluate the performance of the resolution methods

Performance indicators are tools used to evaluate the convergence capacity of PF, which is the set of non-dominated solutions obtained by applying the resolution method, towards the true Pareto Front called (PF_{true}). They are also employed to analyze the distribution of the solutions obtained in the objective space. The most common indicators are listed below [76]:

- **Generational Distance (GD)**: this measure assesses the proximity of the solutions of PF to PF_{true} by calculating the Euclidean distance between each non-dominated solution and its nearest solution in PF_{true}. A low Generational Distance value indicates better convergence. The Generational Distance is computed using Equation 3.11.

$$GD = 1/|PF_{true}|(\sum_{y_1 \in PF_{true}} (\min_{y_2 \in PF} ||y_1 - y_2||)^p)^{1/p} \tag{3.11}$$

  Generally, p=2.

- **Inverted Generational Distance (IGD)**: it is derived from Generational Distance. This indicator assesses both the convergence and the distribution of PF. A low IGD value reflects good overall performance of the utilized algorithme. IGD can be calculated by applying Equation 3.12.

$$IGD = 1/|PF|(\sum_{y_2 \in PF} (\min_{y_1 \in PF_{true}} ||y_1 - y_2||)^p)^{1/p} \tag{3.12}$$

  Généralement, p=2.

- **Hypervolume (HV)** : this measure quantifies the volume of the objective space dominated by the solutions of PF. It evaluates both their convergence and distribution. A high HV value corresponds to good performance of the resolution method. HV value is obtained by utilizing 3.13 written below:

$$HV(PF) = \int_{\mathbb{R}^m} \text{Volume}(\text{dominated by } PF \cap Q)\, dQ \tag{3.13}$$

  Where :

  - m is the number of objectives and $\mathbb{R}^M$ represents the m-dimensional objective space;
  - Q denotes the subset of the considered objective space;
  - $Volume(\cdot)$ corresponds to the volume function.

  This integral is usually approximated using various techniques due to the computational complexity of calculating the exact hypervolume. The applied common applied approach consists in discretizing the objective space into small hypercubes and computing the volume as a function of the number of the hypercubes dominated by the Pareto Front.

- **Walking Fish Group Hypervolume (WFGHV)**: it is an improved version of HV. WFGHV is calculated using the Walking Fish Group (WFG) algorithm introduced in [77]. This method allows the fast calculation of the hypervolume, even for problems containing 5 or more objectives. The WFGHV is computed by Equation 3.14:

$$\text{WFGHV}(PF) = \iiint\limits_{y \in \mathbb{R}^m \setminus \bigcup_{i=1}^{N} V_i} dy_1 dy_2 \ldots dy_m \tag{3.14}$$

  Where N is the number of solutions in PF, m corresponds to the number of objectives, V_i designates the volume of the objective space dominated by the i-th solution in PF. In fact, the integral is taken over the entire objective space excluding the grouping of the dominated volumes of all solutions in PF.

- **Normalized Hypervolume (NHV)**: it is calculated when PF_{true} is known. NHV can be calculated from HV or WFGHV (Equation 3.15). The values of the NHV indicator varies from 0 (best value) to 1 (worst value).

$$\begin{aligned}
&\text{NHV}(PF) = 1 - HV(PF)/HV(PF_{true}) \\
&\text{ou} \\
&\text{NHV}(PF) = 1 - WFGHV(PF)/WFGHV(PF_{true})
\end{aligned} \tag{3.15}$$

- **Maximum Spread (MS)**: this indicator shows the dispersion of the solutions. A higher value of MS reflects a wider distribution of PF. It is computed as follows:

$$\text{MS} = \max_{i,j} \sqrt{\sum_{k=1}^{m} (y_{ik} - y_{jk})^2} \tag{3.16}$$

  Where m is the number of objectives, y_{ik} and y_{jk} represent respectively the k^{th} objective value of i^{th} and j^{th} solutions in PF.

- **Spacing Metric (SP)**: it measures the distance variance between the neighboring solutions in the found PF. A low SP value corresponds to a better distribution of the solutions. Equation 3.17 is applied to calculate MS:

$$\text{SP} = \frac{1}{N} \sum_{i=1}^{N} d_i \tag{3.17}$$

  Where: N is the number of solutions in the PF being evaluated and d_i denotes the distance between the i^{th} solution and its nearest neighbor on the PF.

- **Diversity (D)**: it evaluates the dispersion of non-dominated solutions in the PF. A high value of D indicates better overall performance of the algorithm. Diversity is generally assessed using various metrics or calculations adapted to a specific problem and goals. For example, a common method to measure diversity is to compute the average distance between solutions. However, the exact equation applied to calculate D depends on the context of the optimization problem and the metric chosen to evaluate diversity.

3.2.6 Comparison and review of the recent methods applied to solve optimization problems

Research aiming to improve the performance of the approaches used to solve optimization problems is constantly evolving. In this context, Table 3.1 compares the recent work that have proposed optimization algorithms.

Table 3.1: Comparison and review of recent methods for solving optimization problems

Paper	**Year**	**Type**	**Nature**	**Approach/Objectives**	**Techniques**	**Evaluation problem(s)**	**Number of objectives**	**Results**	**Disadvantages**
[75]	2023	Hybrid	Multi-objective	EMOPSOC	MOPSO and Machine Learning techniques for clustering	Planning and resource allocation in fog computing environments	3	IGD, Hypervolume: 80% increase	Algorithm specific to a particular type of problems; Increased computational complexity; Difficulty in selecting suitable clustering method
[78]	2023	Metaheuristic	Multi-objective	A new algorithm for multi-objective optimizations of real building designs	Bayesian probabilistic approach	BNH problem [79]; Two-Bar truss design [80]; Gear Train Design [81]	2	Fraction of real solutions from PF found: 31.58%; Percentage of suboptimal solutions identified by the algorithm: 8.75%	Algorithm requires parameter tuning for optimal performance; Applicability to other problem types needs to be verified; High computational cost
[82]	2022	Particle Swarms	Single-objective	QSPSO: combining principles of QPSO and solitons concept	Solitons	Benchmark test functions (F1-F21)	1	Success count: 16-20 and computation time: 0.81s-3.25s	Algorithm uses multiple parameters that need to be tuned; Higher computational cost due to additional calculations for quantum mechanical and solitons components
[83]	2022	Hybrid	Multi-objective	Ant-Colony Non-dominated Sorting Genetic Algorithm III (AcNSGA-III): a new hybrid algorithm to solve routing problem in indoor IoT network	NSGA-III and ACO [84]	DTLZ problems and routing in IoT network	6	IGD: 0.115-9.624; HV: 0.901-0.982	Applicability to other network types would require further investigation; Dependence on specific parameters and their tuning for optimal performance
[58]	2021	Bio-inspired	Single-objective	I-GWO: an enhanced version of GWO	DLH	Pressure vessel problem, Welded beam problem, and IEEE 30-bus test system	1	Overall efficiency: 94.2%	Additional complexity compared to GWO may make the algorithm harder to implement; Selection of good parameters can be difficult and requires trial and error

Continued on next page

Table 3.1 – continued from previous page

Paper	Year	Type	Nature	Approach/Objectives	Techniques	Evaluation problem(s)	Number of objectives	Results	Disadvantages
[85]	2021	Metaheuristic	Single-objective	Arithmetic Optimizer Algorithm (AOA): inspired by the distribution behavior of arithmetic operators such as addition, multiplication, division, etc., in solving arithmetic problems.	Arithmetic operators	Benchmark test functions (F1-F13), 5 engineering design problems	1	Ranking based on average fitness value obtained: 1 for 12 functions	High computational cost due to additional calculations required for arithmetic operations; AOA tested on a limited set of reference problems with a small number of variables.
[74]	2021	hybrid	Single-objective	GA-TS: a hybrid algorithm to solve the FSSP	genetic algorithm and tabu search	FSSP	1	Average error: 3.05%; percentage increase of makespan: 14.66%	The algorithm is specifically designed to solve the FSSP; It has been tested on a limited set of FSSP instances; High computational cost due to the combination of two metaheuristics.
[86]	2021	Evolutionary	Multi-objective	MaOEADRA: an algorithm based on dominance and decomposition with reference point adaptation to select solutions for the next generation	reference point adaptation strategy	benchmark test functions: DTLZ, WFG [4], IDTLZ [87], MaF [88], MLDMP [89], and MPDMP [90]	up to 15	HV: best in 81.37% of cases; IGD: best in 84.39% of cases	Performance is not evaluated on a problem other than benchmark problems; Choice of reference points and decomposition strategy can influence performance; High computational cost.
[57]	2019	MIQP	Single-objective	An exact algorithm to efficiently solve realistic Partial Set Covering Location Problem (PSCLP) and Maximal Covering Location Problem (MCLP).	branch-and-Benders-cut	MCLP and PSCLP	1	Average execution time: 0.35s for PSCLP (100000 instances) and 26.08s for MCLP (100000 instances)	The method requires a good initial solution, which can be difficult to obtain for large-scale problems; Performance needs to be verified on other optimization problems.
[72]	2019	Simulated Annealing	Single-objective	OTSA: an improved version of SA inspired by ancient metallurgy techniques	Orikaeshi Tanren	benchmark test functions, engineering optimization problems	1	Percentage of best solutions generated by OTSA: 75%, average execution time: 0.78s	Adjustment of multiple parameters is required, which can be time-consuming and costly in terms of computation; Complexity of the algorithm.

Synthesis The study of recent works conducted to improve the resolution of the optimization problems proves that proposed methods suffer from some weaknesses. First, many solution techniques were designed to deal only with specific types of problems, Which requires to demonstrate their effectiveness and good performance on a wider range of optimization problems. Additionally, some developed methods are only tested on benchmark problems and, therefore, their effectiveness was not validated in real-world scenarios. Furthermore, most existing proposals showed good performance only when applied to solve a small number of objectives. Besides, in many resolution algorithms, the values of their parameters need to be adapted, which can significantly affect the quality of the obtained results. Thus, more research efforts should be made to develop solution methods that are more efficiently applied to deal with both benchmark and real-world problems while handling a large number of objectives.

3.3 Deep Learning techniques

ML, a subfield of artificial intelligence, uses algorithms that enable computer systems to detect complex patterns and structures in data, allowing them to make decisions or predictions and improve their performance over time. DL is an advanced ML technique based on artificial neural networks composed of multiple layers. Unlike the traditional ML methods, which often require manual feature engineering, DL can automatically learn from raw data, making it extremely powerful tool to model complex and unstructured data. Therefore, DL techniques can be applied to a wide range of applications, the most important of which are described in the following subsection.

3.3.1 Main application areas of Deep Learning

DL techniques are generally used to solve various problems in different domains including:

- Regression: prediction of one or more continuous variables as a function of other input variables;
- Classification: consists of assigning labels or categories to data based on their characteristics;
- Image segmentation: partitioning an image into several segments or regions for subsequent analysis and processing;
- Object detection: identification and localization of specific objects in an image or video sequence.
- Activity recognition: identification and classification of specific activities from video sequences;

- Content generation: creating automatic contents such as generating text, image or music.
- Translation: automatic translation between different modes of communication, such as translating text into sign language;
- Graph analysis: extracting patterns and important information from graph data such as social networks or transportation networks;
- Automatic feature selection: automatic identification of the most informative or discriminative features in a data set;
- Enhancement of the image quality.

We are interested, in the rest of this section, in the application of DL techniques to solve regression problems.

3.3.2 Deep Learning for regression problems

Regression in the context of DL refers to a supervised learning task performed to predict continuous target variables from a set of input variables. In this application, DL models are trained on a dataset containing input variables and associated target values. The model learns from these examples by adjusting its internal parameters to minimize the prediction error between its estimated outputs and the actual target values. Once trained, the model is able to accurately predict the target value for new input data. The application of DL in regression typically involves the following steps:

1. Data collection: consists in collecting a dataset containing examples with input variables and continuous target variables to predict;
2. Data preprocessing: it may include steps such as cleaning data to remove outliers or missing values, normalizing features to scale data and dividing data into training, validation and testing sets;
3. Model construction: creating a neural network containing one or more hidden layers. The size and complexity of the model varies according to the nature of the problem and the amount of the available data;
4. Training of the model: the model is, then, trained on the training dataset using an optimization algorithm such as Stochastic Gradient Descent (SGD) or its variants. During training, the model adjusts its weights and biases to minimize a loss function used to measure the overfitting between the predictions made by the model and the actual values of the target variable;

5. Model validation: the performance of the trained model is evaluated on a separate distinct validation dataset to assess its performance and detect any overfitting;

6. Model testing and assessment: the model is evaluated on an independent test dataset to estimate its performance in real-world conditions, which makes it possible to check whether the model generalizes well to new data and to compare its performance with that of other models or approaches.

By following these steps, applying DL in regression can produce accurate models that can adequately predict continuous variables from the input data. These techniques are based on several theoretical foundations presented in the following subsection.

3.3.3 Theoretical foundations of DL

This section presents, in a simplified way, the theoretical foundations of DL techniques.

3.3.3.1 Artificial neurons and neural networks

Artificial neurons and neural networks are the basic elements of DL:

- **Artificial neuron**: it is a fundamental processing unit having multiple weighted inputs, combines them linearly, applies an activation function to the weighted sum and produces an output. The latter can be utilized as the input of other neurons or as the final result of the model. Figure 3.7 exposes a simplified representation of an artificial neuron. Let $x = (x_1, x_2, ..., x_n)$ be the input vector, $w = (w_1, w_2, ..., w_n)$ be the vector of weights associated to each input, b the bias and f the activation function. The artificial neuron calculates the output y as follows:

$$y = f(\sum_{i=1}^{n} w_i \cdot x_i + b) \tag{3.18}$$

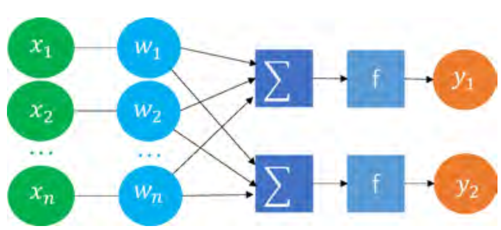


Figure 3.7: A simplified representation of an artificial neuron

- **Neural network**: it is a structure composed of several layers of connected neurons organized into input, output and intermediate layers called hidden layers. Each neuron at the layer is linked to all neurons at the previous layer and the next layer. Outputs in a neural network are calculated iteratively by propagating the inputs through the network to the output layer. Let $x^{(l)}$ be the input vector of layer l, $w^{(l)}$ the weight matrix associated to layer l , $b^{(l)}$ the bias vector associated to layer l and $f^{(l)}$ the activation function of layer l . The output $y^{(l)}$ of layer l is computed as follows:

$$y^{(l)} = f^{(l)}(w^{(l)} \cdot y^{(l-1)} + b^{(l)}) \tag{3.19}$$

  The output values $y^{(l)}$ are, then, used as the inputs of the next layer until obtaining the final output. Figure 3.8 represents a simplified view of a two-layer neural network.

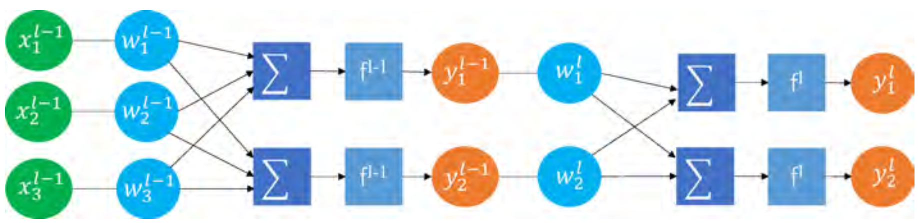


Figure 3.8: A simplified view of a two-layer neural network

3.3.3.2 The functioning of the training and parameters optimization

(a) The functioning of the training: in a DL model, there is a set of trainable parameters that are adjusted and optimized over several iterations in order to minimize the loss function (error) during the training steps shown in Figure 3.9. Each iteration is called an epoch. During each epoch, the training process iterates through the entire data set and, therefore, updates the network weights to minimize the loss function typically used to evaluate the difference between the actual values of the target outputs and their predicted values. During training, the model's performance is usually assessed on a separate dataset called a validation set, which is not employed for the training itself. In fact, validation is carried out to control the model's ability to generalize correctly to new data, which allows estimating how well the model performs on data it has not yet seen. It is often conducted at regular intervals (e.g., at the end of each epoch). For this reason, the number of epochs is a crucial hyperparameter which influences the training process. Another hyperparameter that should be adequately chosen is the batch size. Indeed, to be efficiently processed, data must be divided into batches of fixed size. A batch is a subset of the training data used to optimize and update the network weights at each iteration. The weights of the model to be trained are generally initialized with random values. Then, at the end of processing each batch, they are updated to minimize the loss function by applying an optimization algorithm. In the following part, we will describe the most commonly applied optimizers.

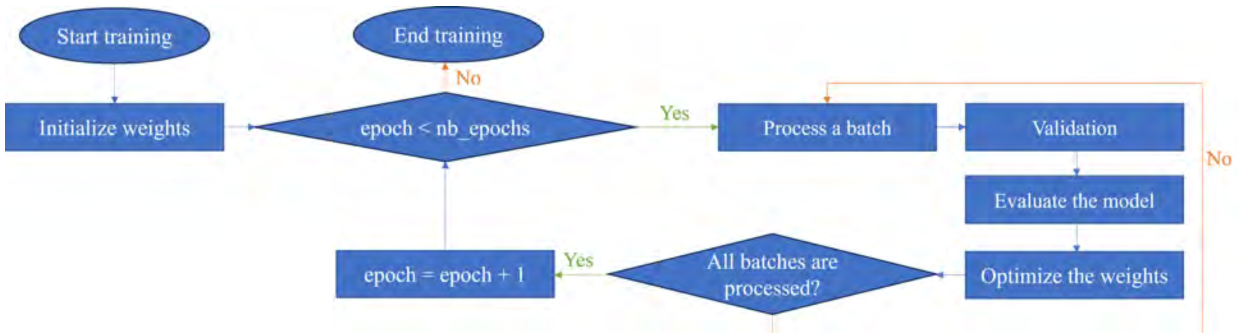


Figure 3.9: Training flowchart of a Deep Learning model

(b) **Optimization and parameters adjustment techniques**: To achieve the optimal performance of a DL model, the weights of the neural network used should be optimized. Afterwards, the most widely utilized optimizers are described. w are the set of weights to optimize, L is the loss function, α corresponds to the learning rate and $\nabla L(w_t)$ denotes the gradient of the loss function with respect to the parameters w evaluated at iteration t.

- The Gradient Descent (GD): it is employed to update the parameters (the weights) in the direction opposite to that of the loss function gradient with respect to the parameters. Mathematically, the update of the parameters w at iteration $t+1$ is given by :

$$w_{t+1} = w_t - \alpha \nabla L(w_t) \tag{3.20}$$

  The learning rate (α) controls the size of the steps of the parameter update. Indeed, very reduced learning rate can lead to slow convergence. However, very high learning rate may result in instability and divergence of the learning process.

- The SGD: is a variation of GD that uses a small batch of data at a time to update the weights of the model. At each iteration t, a training example (x_i, y_i) is randomly sampled from the training dataset. The model parameters are, then, updated applying Equation 3.21:

$$w_{t+1} = w_t - \alpha \nabla L(w_t; x_i, y_i) \tag{3.21}$$

  SGD is one of the widely adopted optimization algorithms utilized to train DL models due to its easy implementation and high efficiency [91].

- The Batch Gradient Descent (BGD) [92]: is a variant of GD that employs the entire batch data set to calculate the gradient of the loss function and update the model parameters. Mathematically, the update of the weights w at iteration $t+1$ with the BGD is done using Equation 3.22:

$$w_{t+1} = w_t - \alpha \frac{1}{|B|} \sum_{i \in B} \nabla L(w_t; x_i, y_i) \tag{3.22}$$

where $|B|$ is the batch size.

It is worth to note that BGD is more accurate than SGD in optimizing the weights, but it can be slower than it as BGD calculates the gradient over the whole batch data set.

- The Follow the Regularized Leader (FTR): employs a regularized loss function $L_t(w)$ which combines the loss function $\ell_t(w)$ with a regularization penalty $R(w)$ on the model weights: $L_t(w) = \ell_t(w) + R(w)$. The weights are updated using Equation 3.23:

$$w_{t+1} = w_t - \alpha_t \nabla L_t(w_t) \tag{3.23}$$

  Combining GD optimization with regularization to update model weights makes FTR effectively utilized to handle large datasets [93].

These optimization algorithms can be combined with several DL methods to solve regression problems. The next subsection presents the most intensively applied methods.

3.3.4 DL methods used to solve regression problems

To solve regression problems in DL, numerous methods can be used, including:

- **Fully-Connected Neural Networks**: it is a network containing many fully-connected layers. Each neuron in a given layer is connected to all neurons in the previous layer and next one.

- **Deep Neural Network (DNN)**: it is made up of several layers including an input layer, an output layer and intermediate (hidden) layers. DNNs can handle non-linear relationships within data and process large amounts of it.

- **Artificial Neural Network (ANN)**: these computer models, composed of neurons organized in layers, are inspired from the functioning of the human brain.

- **Convolutional Neural Network (CNN)**: they are primarily used for computer vision. They can also be applied to solve regression problems when the inputs have a spatial structure. According to [94], CNNs are the most popular DL technique because of their effectiveness in processing large data.

- **Recurrent Neural Network (RNN)** [95]: connections between nodes can be recursive, showing that the input of some nodes depends on their own previous output. This characteristic makes RNNs able to efficiently process sequential data.
- **Long Short-Term Memory (LSTM)**: it is a version of RNN specially designed to efficiently deal with long-term dependencies in sequences.
- **Gated Recurrent Unit network (GRU)**: a variant of RNN characterized by a reduced number of parameters and rapid convergence. It was developed in [96].
- **Auto-encoder Neural Network (AENN)**: it is utilized to learn compressed representations of input data. They can be also used to reduce dimensionality of data and reconstruct it, which is useful to solve some regression problems.
- **Multi-layer Perceptron (MLP)**: connections are established only from input to output (in one direction). MLP does not include any recurrence, proving that the output of one neuron cannot influence its next input.
- **Ensemble learning**: This approach consists in merging the outputs of two or more models in order to improve the obtained results. For example, some methods computes the mean of the outputs provided by multiple DL and ML models.

The DL technique to employ should be chosen according to the specific requirements of the problem to be solved and the nature of the data in order to solve optimally the regression problems. Several criteria are used to evaluate the performance of DL techniques. The following subsection lists the most commonly utilized criteria.

3.3.5 Criteria used to evaluate the performance of the DL techniques

DL techniques used to solve regression problems can be assessed applying different performance criteria depending on the specific needs of the problem to be solved and the available data. The commonly-used evaluation criteria are described below. In the criteria equations, y_i denotes the actual value, $\hat{y}_i$ represents the value predicted by the DL model and n is the total number of the measurement points.

- Mean Absolute Error (MAE): it is the mean of the absolute values of the differences between the predicted values and the actual ones. MAE measures the mean size of the prediction errors without considering their direction.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.24}$$

- Mean Squared Error (MSE): it refers to the mean of the squares of the differences between the predicted values and the actual ones. MSE penalizes large errors.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.25}$$

- Root Mean Squared Error (RMSE): it is the square root of the mean of the squares of the differences between the predicted and actual values.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.26}$$

- R-squared (R2): it quantifies the amount of the variance of the values predicted by the actual values (Equation 3.27). It varies between 0 and 1, where 1 indicates the perfect fit of the model to the data.

$$R2 = 1 - \sum_i (y_i - \hat{y}_i)^2 / \sum_i (y_i - \overline{y})^2 \tag{3.27}$$

$\overline{y} = \sum_i y/n$

- Mean Absolute Relative Error (MARE): it is the average of the absolute values of the relative differences between the predicted and actual values. It is expressed in percentage.

$$MARE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{|y_i|} \times 100\% \tag{3.28}$$

- Accuracy: represents the number of correct predictions made by the model compared to the total number of predictions.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \times 100\% \tag{3.29}$$

- Recall: it is a performance measure used in classification problems to evaluate the system's ability to correctly capture and identify relevant objects. This measure can be adapted in regression, as demonstrated in Equation 3.30.

$$Recall = nb\_c\_pred/tot\_nb\_succ\_pred \tag{3.30}$$

where nb_c_pred is the number of the correct predictions (the number of correctly- predicted relevant values) and $tot\_nb\_succ\_pred$ represents the total number of relevant values to predict.

- F1-score: It is the mean of accuracy and recall. It provides a unique measure that balances the compromise between these two metrics (Equation 3.31).

$$F1\text{-}score = 2 \times \frac{Accuracy \times Recall}{Accuracy + Recall} \tag{3.31}$$

3.3.6 Comparison and review of the recent studies on optimizing DL techniques by using metaheuristics

Recently, many efforts have been made to optimize the performance of DL techniques applying metaheuristics. Table 3.2 presents recent works addressing this approach.

Synthesis The study of recent works conducted to enhance the performance of DL techniques utilizing metaheuristics shows that they present certain limitations. In fact, some studies focused on using metaheuristics to optimize the hyperparameters of DL models, while others aimed at improving both the hyperparameters and the initial values of the weights of these models. On the other hand, the objective of some researches was to optimize data by reducing its size and selecting the most relevant attributes by using metaheuristics. Despite their importance and efficiency, these approaches are often designed to be applied in specific types of DL techniques, which prevents their generalization. Although the employment of metaheuristics has proven effective in improving the performance of the studied DL models, no previous work has shown the efficiency of metaheuristics to optimize the weights of DL models during training. It is, therefore, necessary to deal with weight optimization utilizing metaheuristics in order to benefit at maximum from their advantages and, consequently, accelerate convergence towards the optimal weight values and improve the performance of the trained models. Furthermore, it is crucial to generalize the proposed optimizers and make them applicable in several DL techniques.

Table 3.2: Comparison and review of recent works optimizing DL techniques using meta-heuristics

Paper	Year	DL technique	Metaheuristic	Optimization objective	Studied problem	Results without optimization	Results with optimization
[97]	2023	CNN	Particle Swarm Optimization without Velocity (PSWV)	optimize hyperparameters	Image classification	-	Error rate: 3.64%
[98]	2023	LSTM	Artificial Rabbits Optimization Algorithm (ARO) [99]	optimize hyperparameters	Stock price prediction	MSE: 44.44, MAE: 5.21, R2: 82.9%	MSE: 24.287, MAE: 3.804, R2: 90.7%
[100]	2022	DNN	Improved Particle Swarm Optimization (IPSO)	optimize hyperparameters	Impact of social distancing on COVID-19 spread	MAE: 6.1002, RMSE: 49.3345	MAE: 4.8177, RMSE: 45.0471
[101]	2022	CNN+LSTM	Whale Optimization Algorithm (WOA)[102]	optimize hyperparameters	Emotion recognition in speech	Accuracy: 92.24%	Accuracy: 99.47%
[103]	2022	CNN	GWO	optimize hyperparameters	Skin cancer detection	-	Accuracy: 81.46%
[104]	2021	CNN	Binary Particle Swarm Optimization (BPSO)	identify most relevant features and reduce data dimensionality	Intelligent Intrusion Detection System (IDS)	-	Accuracy: 94.3%, Recall: 91%
[105]	2020	DNN	GWO	data reduction: reduce attribute dimensionality by selecting highly influential attributes	Intrusion detection in IoMT (Internet of Medical Things) architecture	Accuracy: 99.2%	Accuracy: 99.9%
[106]	2020	DNN	Genetic Algorithm	optimize hyperparameters	Energy consumption prediction in buildings	RMSE: 11.83	RMSE: 10.29
[107]	2020	ANN	PSO	optimize hyperparameters and initial weight values	Prediction of maximum power point of a photovoltaic array	Efficiency: 98.3%	Efficiency: 99.49%
[108]	2019	CNN	cPSO: a new variant of PSO proposed	optimize hyperparameters	Ablation analysis and comparison with similar works	-	Error rate: 8.67%

3.4 Comparison and review of the recent studies conducted to solve routing problems in IoT networks using optimization algorithms

Many studies have focused on enhancing the routing performance in IoT networks using meta-heuristics. In this context, Table 3.3 presents a review of the recent research works dealing with this issue. In order to avoid repeating the descriptions of the works already exposed in Table 2.2 (the section 2.3.3), they are simply referenced in this table. The latter includes details about the applied optimization algorithms and the results they provided. It also highlights their importance in improving the performance of routing in IoT networks.

Synthesis As optimization is widely and effectively applied in many fields, it is necessary to benefit from this success to study routing in IoT networks. In fact, this approach can improve the performance of this task and reduce its complexity. The review of the previous research works highlights the importance of metaheuristics in enhancing the routing performance in IoT networks. However, despite this progress, routing in IoT networks presents many challenges that affect the network overall performance. Each of the previously-mentioned protocols focuses on a limited number of routing requirements, which demonstrates that none of them are entirely effective in solving all routing problems. Additionally, the previous studies have some drawbacks. In fact, most performed experiments are often limited to simulations and most existing approaches are not based on simulators specific to IoT networks, which affects the validity of their results. Moreover, these works often have a limited number of objectives. However, in a context where IoT applications increasingly require better performance and more rapid communication speed, especially in real-time applications, it is imperative to design a more robust and efficient IoT protocol. It was proven that the use of the optimization algorithms enhances the performance and efficiency of routing strategies applied in IoT networks. Therefore, it is crucial to benefit from their efficiency to solve complex multi-objective problems in order to develop new routing solutions and, therefore, to improve its performance in IoT networks.

Table 3.3: Comparisons between recent works solving the routing problem in IoT networks based on optimization algorithms

Paper	Year	Type	Approach/Objectives	Optimization algorithm	Results	Disadvantages
[15]	2.2					
[17]	2.2					
[109]	2022	Geographic	Selecting routes considering node positions, their energy level, and traffic data	Bacteria Foraging Optimization (BFO) [110]	Delay: reduced by 12%; Throughput: increased by 10%; Residual energy: increased by 11%; PDR: increased by 7 to 9%	Not considering mobile nodes; The algorithm assumes nodes know their own location as well as that of their neighbors, which is not always true in real scenarios; Simulations with Matlab
[83]	2022	QoS-based	Developing an energy-efficient routing algorithm	AcNSGA-III: a proposed new optimization algorithm	Average latency: 1.3s-2.8s; Average lifetime: ≈ 19400 s	No comparisons to other existing routing solutions; Applicability to other types of networks would require further investigations.
[111]	2022	Reactive	Minimizing the maximum average transmission distance between nodes while balancing energy management	MOGWO	PDR: 98%; Delay: reduced by 40%; Energy consumption: reduced by 30%	No mobile nodes; No real-world experiments; Additional delays in data transmission; Higher computational load and longer convergence time
[112]	2022	QoS-based	Basing routing on a novel optimization algorithm and a new fitness function	Reposition PSO (RPSO): a proposed new variant of PSO	Network lifetime: 14% to 29% higher; Number of inactive sensor nodes: reduced by 52% to 58%; Energy consumption: reduced by 61% to 70%; Throughput: increased by 13% to 36%	Comparison with other optimization algorithms rather than with other existing routing solutions; No real-world experiments
[113]	2021	Multi-path	Designing a new hybrid optimization algorithm by integrating Sun Flower Optimization (SFO) [114] and GWO to establish multi-path routing	SFO and GWO	Delay: 0.779 s; Network lifetime: 98.039%; Throughput: 47.368%	No real-world experiments; Only a single objective function;
[115]	2021	Multi-path	Computing fitness function using distance between nodes in the network	Genetic algorithm	Energy consumption: 0.9-5.9J; Active relay node rate: 30%	No comparisons with other routing solutions; Focus only on energy consumption; No real-world experiments
[116]	2020	Data-driven	Solving multimedia data routing in an IoT network	Shuffled Frog Leaping Optimization Algorithm (SFLA) [117]	PDR: ≈ 90%; Average throughput: ≈ 3.5 Mbps; Lifetime: 184 cycles	No real-world experiments; The proposal does not address data redundancy, a challenge that significantly impacts the performance of a multimedia network

Continued on next page

Table 3.3 – continued from previous page

Paper	Year	Type	Approach/Objectives	Optimization algorithm	Results	Disadvantages
[118]	2020	QoS-based	Developing a routing algorithm for Mobile Ad-hoc Networks	Intelligent Water Drop Optimization Algorithm (IWDOA) [119] and FOA [120]	PDR: 82.64% for IWDOA and 82.26% for Firefly Optimization Algorithm (FOA)	No real-world experiments; Does not support self-stabilization (acting automatically in case of issues) and malicious node identification, both critical challenges in IoT-MANET networks [121]
[122]	2020	QoS-based	Developing a routing protocol for heterogeneous wireless sensor networks	Proposed variant of GWO	Network lifetime: 1400-2300 cycles; Throughput improvement of 94.02%	No real-world experiments; Considers only a single objective, energy consumption; The studied environment contains only communication between a base station and star heads; No mobile nodes.
[123]	2020	Data-driven	Developing Content Awareness Routing Algorithm for IoT Networks (CARA-IoT), a routing algorithm in Multimedia IoT networks	ACO	PDR: 70%-80%	No real-world experiments; Does not address the data redundancy issue which represents a significant challenge in Multimedia networks
[124]	2020	QoS-based	Selecting routes based on residual energy and trust level for secure and reliable routing	PSO and Salp Swarm Optimization (SSO) algorithm [125]	Delay: 0.08ms; Detection rate: 83%	No real-world experiments
[126]	2019	QoS-based	Reducing the probability of hacking	Imperialist Competitive Optimization Algorithm (ICOA) [127]	Confidentiality: 0.72; Integrity: 0.70; Availability: 0.43	No real-world experiments; Tests conducted in a small-scale mesh network (20 nodes); Presented results do not include comparisons with other protocols.

3.5 Comparison and review of the recent studies conducted to solve indoor localization problems in IoT networks using metaheuristics and/or DL methods

In the literature, several studies applied metaheuristics and DL techniques to develop indoor localization solutions in IoT networks. Table 3.4 presents some details about these works. To avoid repeating the descriptions of the studies already shown in Table 2.3 (the section 2.4.2), they are referenced in the table below. The latter highlights the used existing metaheuristics and/or DL techniques as well as the obtained results to illustrate their contribution to improve the efficiency to solve indoor localization problems.

Synthesis Recent research works have examined the use of metaheuristics and/or DL techniques to develop indoor localization solutions in IoT networks. The study presented in this section shows that they differ in their methodological approach. More precisely, some works have utilized either metaheuristics or DL techniques, while others have employed hybrid techniques combining the two types of methods. This diversity demonstrates the good potential of the metaheuristics and DL techniques to solve complex indoor localization problems and to obtain better results in this domain. However, these approaches have some limitations, as proven in 2.4.3. First, most of the previous studies are based on simulated experiments rather than real data, which reduces the validity of their findings. Additionally, the widespread use of RSSI measurements, which are sensitive to environmental disturbances, such as interference and signal attenuation, poses a major challenge. Despite the advantages and effectiveness of the approaches based on metaheuristics and DL, they cannot meet the growing requirements of IoT networks in terms of performance and location accuracy. Thus, it is imperative to explore new approaches that can fully leverage the combined benefits of these two promising paradigms while using less sensitive metrics than RSSI.

Table 3.4: Comparisons between recent works solving the indoor localization problem in IoT networks using metaheuristics and/or DL methods

Paper	**Year**	**Metaheuristic**	**DL**	**Type**	**Approach/Objectives**	**Techniques**	**Technology**	**Measurements**	**Results**		**Disadvantages**
									MAE	**Accuracy**	
[43]	2023	✗	✓	2.3							
[31]	2023	✓	✗	2.3							
[32]	2023	✓	✗	2.3							
[44]	2023	✗	✓	2.3							
[128]	2023	✓	✗	range-based	Developing a solution for target localization and tracking	PSO	-	RSSI	-	93.09%	No comparisons with other existing solutions; No real-world experiments
[46]	2022	✗	✓	2.3							
[129]	2022	✓	✗	range-based	Solving the localization problem for Edge computing nodes	GWO and Moth-Flame Optimization (MFO) [130]	-	RSSI	<1m for GWO and <2m for MFO	-	Comparison with other metaheuristics rather than existing localization solutions; RSSI measurements may be affected by environmental factors (noise or interference); No real-world experiments
[131]	2022	✓	✗	range-free	Proposing a hybrid optimization algorithm and integrating it with the DV-HoP algorithm	Tunicate Swarm Algorithm [132] and Harris hawk optimization [133]	-	-	0.45 m	-	Hybridizing two metaheuristic algorithms may increase the complexity of the localization algorithm; Tuning parameters may prove challenging and time-consuming; No real-world experiments (Matlab)
[134]	2022	✓	✗	range-based	Developing a hybrid approach combining the strengths of the enhanced PSO, Inertial Measurement Unit (IMU), and RSSI fingerprinting	Enhanced version of PSO proposed	Wi-Fi	RSSI	0.7m	-	Complexity of combining 3 methods; RSSI measurements may be affected by environmental factors; Solution requires precise map information (which can be challenging and time-consuming)
[48]	2021	✗	✓	2.3							
[49]	2021	✗	✓	2.3							

Continued on next page

Table 3.4 – continued from previous page

Paper	Year	Metaheuristic	DL	Type	Approach/Objectives	Techniques	Technology	Measurements	Results		Disadvantages
									MAE	Accuracy	
[135]	2021	✓	✗	range-based	Iteratively updating positions of unknown nodes based on their distances to anchors until positions converge to a local minimum	Group Teaching Optimization Algorithm (GTOA) [136]	-	RSSI	-	64.5-94%	Comparison with other metaheuristics rather than existing localization solutions; RSSI measurements may be affected by environmental factors; No real-world experiments (Matlab)
[55]	2020	✗	✓	2.3							
[137]	2020	✓	✗	range-based	Develop a localization solution for smart parking systems: minimize localization error and reduce the required number of anchors	MOGWO	-	RSSI	RMSE $\approx$ 0.42-0.52m	67%-93%	Not taking into account a dynamic environment (mobile vehicles); No real-world experiments; RSSI measurements may be affected by various factors

3.6 Conclusion

In conclusion, this chapter has explored two crucial areas offering valuable perspectives on solving complex problems. We examined the different types of optimization problems, the associated solution methods and the performance criteria used to evaluate these methods. We also explored the field of Deep Learning, examining its theoretical foundations, practical applications and associated performance criteria. Finally, we highlighted the convergence between optimization and DL, studying works that use metaheuristics to optimize DL techniques, thus paving the way for more efficient solutions to solve a variety of real-world problems. Reviews regarding the application of these beneficial techniques to solve routing and localization problems in IoT networks have been studied. To sum up, this chapter provides a comprehensive overview of the concepts, methods and research trends in the fields of optimization and DL. Therefore, it presents a foundation for further contributions to solve complex routing and location problems in IoT networks.

Part II

Contributions

Chapter 4

Theoretical contributions

Contents

4.1 Introduction

This chapter provides a detailed presentation of our theoretical contributions including a mathematical modeling of routing in IoT networks as well as an improved version of MOGWO algorithm. It also describes the optimization of the parameters of DL models using metaheuristics. Furthermore, a section is devoted to illustrate our proposal about indoor localization in IoT networks. This proposal combines DL techniques and metaheuristics. Taking advantages from previous approaches, we conclude this chapter by exposing our hybrid, multi-objective geographic routing model to be applied in mesh-architectured IoT networks.

4.2 Mathematical modelling of routing in IoT network

In this section, we propose a mathematical modeling of routing in an IoT network. This modeling allows determining the structure and characteristics of the IoT network considered in our contributions. It also helps to formalize the concepts and relationships between the objects connected in this network, which makes the routing developed in the following sections more structured and facilitates the understanding of their functioning.

4.2.1 Notation

The suggested mathematical modeling considers the following sets, decision variables and parameters:

- N: The considered network.
- O: The set of objects connected in N.
- C: Connections between the connected objects.
- $N(o)$: The neighbors of object o.
- $N2H(o)$: The 2-hop neighbors of object o.
- s_o: The initial source of a transmission.
- d_o: The final destination of a transmission.

- c_o: The current object during routing (where the message arrives during routing).
- n_o: A neighboring object of c_o.
- I: A unique identifier of each object o in the network.
- (o_x, o_y): The coordinates of an object o.
- (s_x, s_y): The coordinates of the initial source s_o.
- (d_x, d_y): The coordinates of the final destination d_o.
- $R(o)$: A degree of reliability assigned to each object o in the network.
- $N_T(o)$: The neighboring table of an object o.
- $Distance(o_1, o_2)$: The distance between two neighboring objects o_1 and o_2.
- ref_{dist}: A reference distance between two neighboring objects.
- E_{init}: The initial energy of each object.
- E_{T}: The transmission energy consumption.
- E_{R}: The reception energy consumption.
- E_{Sleep}: The sleep energy consumption.
- E_{Listen}: The listening energy consumption.
- E_{residual}: The residual energy of each object.
- $E^i_{\text{T\_bit}}$: The energy consumed by an object i when it emits a bit.
- $E^i_{\text{R\_bit}}$: The energy consumed by an object i when it receives a bit.
- S_m: The size of a message to send in bits.
- S_{cm}: The size of a control message to send in bits.
- $Nb\_hops(routing\_path)$: The number of hops in a routing path.
- R: A route which is a list of object identifiers in the network.
- $DLModel$: is the trained Deep Learning localization model.
- S_i: a star i.

- H_i: the head of the star i.
- G: a gateway.

4.2.2 Objectives

The objective functions are generally applied in IoT routing problem to optimize the overall cost of each route and select the best path from the available ones. This cost is calculated by considering the distance between objects traveled during routing, energy performance, the traffic state which will influence latency and other relevant metrics. The objectives to consider during this multi-hop routing are described in the following subsections.

4.2.2.1 The positive progress towards the final destination

- Objective: This function aims at obtaining the greatest positive progress towards the final destination d_o, starting from the current object c_o, passing through a neighboring object n_o and getting closer to d_o (Figure 4.1). The criterion used to evaluate this progress is the percentage of advancement towards the destination.
- Objective function:

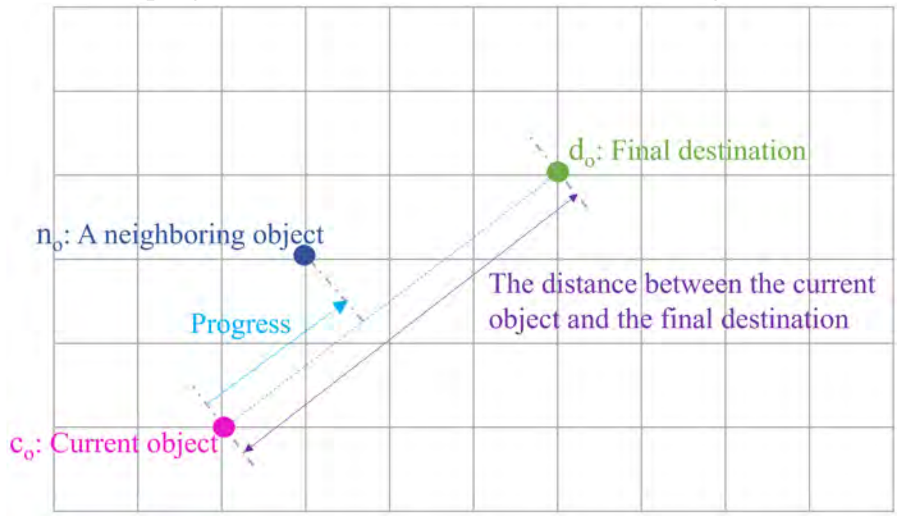
$$\begin{aligned} &Maximize \frac{Progress \times 100}{D(c_o, d_o)} \\ &\text{Where } Progress = \frac{D(c_o, d_o)^2 + D(c_o, n_o)^2 - D(n_o, d_o)^2}{2 \times D(c_o, d_o)} \end{aligned} \tag{4.1}$$

D is a function that returns the distance between two objects considered as parameters.

Demonstration : By applying the Pythagorean theorem to the lengths of the sides in a right triangle. : $D(c_o, n_o)^2 - Progress^2 = D(n_o, d_o)^2 - (D(c_o, d_o) - Progress)^2$
That gives : $2 \times D(c_o, d_o) \times Progress = D(c_o, d_o)^2 + D(c_o, n_o)^2 - D(n_o, d_o)^2$

Example Let $(6, 4)$, $(2, 7)$ and $(12, 17)$ be the coordinates of c_o, n_o and d_o, respectively. $D(c_o, d_o) = 5$
$D(c_o, n_o) = 2.24$
$D(n_o, d_o) = 3.16$
Therefore, $Progress = \frac{5^2+2.24^2-3.16^2}{2\times5} = 2.0032$

Figure 4.1: Positive progress towards the final destination starting from the current object



4.2.2.2 Efficient management of energy

In order to efficiently manage energy, the following objectives are taken into account:

Energy consumption

- Objective: The objective of this function is to optimize the average energy consumption of IoT objects when receiving and transmitting messages over a single hop. At each routing step, an intermediate object c_o, located between the initial sender and the final receiver, receives an amount S_m of data before transmitting it to another object, n_o located at a distance d. This process must respect two constraints. On the one hand, the energy consumption of c_o must not exceed its residual energy level. On the other hand, n_o must have a sufficient residual energy to receive and return data to the next c_o when required.

- Objective function:

$$
\begin{aligned}
&Minimize \sum E^i_{\mathrm{R}} + E^i_{\mathrm{T}}(d) \\
&\text{Where} \quad E^i_{\mathrm{R}} = E^i_{\mathrm{R}} + S_m \times E^i_{\mathrm{R\_bit}} \quad \text{and} \quad E^i_{\mathrm{T}}(d) = E^i_{\mathrm{T}} + S_m \times E^i_{\mathrm{T\_bit}} + S_m \times d^2 \\
&\text{Subject to: } E^c_{\mathrm{R}} + E^c_{\mathrm{T}}(d) \leq E_{\mathrm{residual}}(c_o) \quad \text{and} \quad E^n_{\mathrm{R}} + E^n_{\mathrm{T}}(d) \leq E_{\mathrm{residual}}(n_o)
\end{aligned}
\tag{4.2}
$$

Standard-deviation of energy

- Objective: This function aims at minimizing the Standard-Deviation of Energy (SDE), which represents the variation in the total energy consumed in the IoT network. It shows how the amount of energy consumed in the network differs from one object to another, making it possible to assess the balance of energy consumption in the network. In fact, a small variation indicates a balanced distribution of energy, while a higher variation shows differences in the amount of energy consumed by each object.

- Objective function:

$$\begin{aligned} &Minimize\ SDE(N) \\ &\text{Where} \quad SDE(N) = AVG(VAR(\sum E_{\text{T}} + E_{\text{R}} + E_{\text{Sleep}} + E_{\text{Listen}})) \end{aligned} \tag{4.3}$$

Energy efficiency

- Objective: energy efficiency, in the introduced modeling, consists in effectively managing the energy remaining in objects. It involves minimizing the percentage of energy consumed by an object to receive and retransmit the message compared to its residual energy. The criterion to use is the percentage of energy to be consumed in relation to the residual energy.

- Objective function:

$$\begin{aligned} &Minimize\ \frac{(E^c_{\text{R}} + E^c_{\text{T}}(ref_{dist})) \times 100}{E_{\text{residual}}(n_o)} \\ &\text{Where} \quad E^c_{\text{R}} = E^c_{\text{R}} + S_m \times E^c_{\text{R\_bit}} \text{ and } E^c_{\text{T}}(ref_{dist}) = E^c_{\text{T}} + S_m \times E^c_{\text{T\_bit}} + S_m \times ref^2_{dist} \end{aligned} \tag{4.4}$$

The network lifetime

- Objective: The network lifetime corresponds to the period during which, at least, one object remains operational in the network. This function is employed to maximize the lifetime of the network. To extend this duration at maximum, it is necessary to fairly distribute the load of the routing communications between objects in order to delay the stopping of the first object as much as possible. This extension can be anticipated by calculating the difference between the initial energy of all objects and the sum of the energy standard deviation and the mean of the energy consumed by the IoT objects.

- Objective function:

$$\begin{aligned} &Maximize \quad (\sum_{o_i \in O} E_{\text{init}}(o_i) - (SDE(N) + C_{avg}(O)) \\ &\text{Where } C_{avg}(O) = \frac{\sum_{o_i \in O} E_{\text{T}}(o_i) + E_{\text{R}}(o_i) + E_{\text{Sleep}}(o_i) + E_{\text{Listen}}(o_i))}{|O|} \end{aligned} \tag{4.5}$$

4.2.2.3 Global reliability

- Objective: The overall reliability consists in maintaining reliable connections between objects, ensuring efficient and continuous communications between them. In the proposed modeling, reliability is based on two degrees: a degree of reliability and a degree of confidence for each object. The former is the probability that the sensor will function correctly. However, the latter represents the level of confidence assigned to an object after the previous experiences.

- Objective function:

$$\begin{aligned} & Maximize \sum_{i=1}^{|O|} (r = Reliability_i \times Confidence_i) \\ & \text{Where } Reliability_i = \frac{op\_time(o_i)}{ob\_time} \end{aligned} \tag{4.6}$$

  $Op\_time(o_i)$ is the time during which the object o_i remains operational, and ob_time is the total observation time.

4.2.2.4 Continuity

Local connectivity

- Objective: local connectivity is considered, in this modeling, as the number of neighbors that a node has at a distance of one hop. Better connectivity can ensure continuity of the path towards the final destination by reducing the risk of gaps. The void problem occurs when a packet arrives at an object that has no other neighbors.

- Objective function:

$$Maximize |N(n_o)| \tag{4.7}$$

Advantageous neighbors

- Objective: it aims at having advantageous candidates among the neighbors for the next hop in terms of connectivity, positive progress towards the final destination and degree of reliability.

- Objective function:

$$\begin{aligned} &Maximize|N2H(n_o)| \\ &Maximize \max_{o_i \in N2H} Progress(o_i, d_o) \\ &Maximize \max_{o_i \in N2H} R(o_i) \end{aligned} \tag{4.8}$$

4.2.2.5 Transmission latency

- Objective: Transmission latency is minimized to reduce the time elapsed between the transmission of data from the initial source and its reception by the final destination. In the developed modeling, the transmission latency can be affected mainly by the physical distance between nodes.

- Objective function:

$$\begin{aligned} &Minimize \sum T(s_o, d_o) \\ &\text{Where } T(s_o, d_o) \text{ is the transmission time between } s_o \text{ and } d_o \end{aligned} \tag{4.9}$$

This modeling is used to represent our contributions about routing in an IoT network. First, MOGWO is used to develop reactive and multi-objective routing. Second, a geographic, hybrid and multi-objective routing based on an improved version of MOGWO is proposed. The following section represents the first contribution regarding routing in an IoT network.

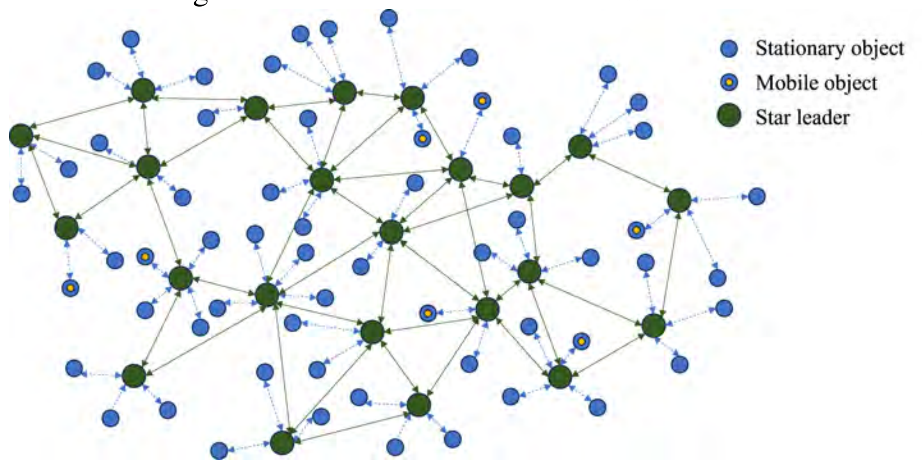
4.3 MOGWO-based routing

In this contribution, we utilize the MOGWO algorithm to model reactive and multi-objective routing in an IoT environment as GWO is one of the recent meta-heuristics which has gained great interest in several fields. Its ability to deal with multi-objective problems in its MOGWO version allows modelling and developing the targeted routing. This section presents the network model, the assumptions to take into account and the routing algorithm to be developed.

4.3.1 The network model

The considered IoT network model is illustrated in Figure 4.2. It is based on a hybrid topology and, thus, uses a star structure. Each star is made up of a specific device operating as its leader acting as the central point for traffic management within the star. This section focuses on multi-objective

Figure 4.2: The considered IoT network model



routing in the IoT network having hybrid topology. It deals specifically with the routing process itself, and, consequently, excludes the formation of stars and the selection of their leaders. Therefore, we assume that the model of the network and its components, such as the stars and their leaders, were previously established. Star leaders maintain communications with the central computer via pre-established paths since these leaders are stationary and do not change position. They regularly send control messages to this central computer containing data about their remaining energy amount and the composition of the stars in case of structure change. The central computer has a global view about the network as well as the links between the star leaders. It uses the information received periodically from the star leaders to update its overall knowledge on the network. The central computer also applies the metaheuristic algorithm, employing this periodically-updated information, to develop communication paths through reactive and multi-objective routing. The considered objectives are those presented in section 4.2.2 related to efficient energy management and transmission latency optimization. When an object in the star wants to transmit a message, it simply sends it to its star leader that asks, subsequently, the central computer to establish an optimized path to the final destination. Then, the central computer initiates a multi-hop routing process between the different star leaders in the network using the MOGWO multi-objective optimizer.
The multi-hop route begins with the star leader containing the initial source and ends by the star leader including the final destination if located in the IoT network. Otherwise, if the destination is outside it, the route continues until reaching an appropriate gateway to be transmitted outside. Thus, this network model offers a flexible and scalable communication infrastructure within the IoT environment. It not only allows data exchanges between objects (Machine to Machine (M2M)), but also with data collection and processing points internal to the network. In addition, it facilitates interactions with the external entities, promoting the integration of the IoT network with other systems and applications.

4.3.2 Assumptions

The following assumptions are taken into account in this research work:

- Star leaders have a greater amount of initial energy than other objects that constitute the stars. These leaders are stationary (do not change positions).
- The other objects are homogeneous, which means that they have the same amount of initial energy and the same transmission range.
- The network contains fixed objects and mobile ones.
- Wi-Fi is the communication technology used in the network. In fact, the Wi-Fi network is configured in ad-hoc mode, permitting direct communication between different objects without requiring centralized infrastructure.
- Each star leader must maintain periodical communications with the central computer, allowing it to have an up-to-date global view about the network, the energy levels and the availability of the communication routes. This mechanism also guarantees a responsiveness and an efficiency of routing when transmitting data across the network.

4.3.3 Routing algorithm

Consider a scenario where a communicating object s_o, part of the star S_s, wants to transmit a message m to a destination D. The latter can be another object d_o belonging to a distinct star S_d or a gateway B leading to an external network. H_s is the leader of the star S_s, while H_d corresponds to the leader of S_d. OC is the central computer responsible for routing.
To resolve message routing, we assume that:

- Routing is considered an optimization problem;
- Star leaders represent the solutions in the search space;
- The routing to be carried out consists of multiple hops between star leaders;
- At each step of solving the optimization problem, the next hop (i.e., the next leader to reach) is determined;
- The solution returned by the MOGWO optimization algorithm is the routing path;
- The objective functions applied to ensure efficient energy management and transmission latency reduction, described in section 4.2.2, are taken into account.

Algorithm 8: MOGWO-based routing algorithm

Result: The route R
Transmit(m, s_o, H_s)
if $d_o \in S1$ **then**
    Transmit(m, H_s, d_o)
else
    Transmit(routing_request, H_s, OC)
    $X_{current} \leftarrow H_s$
    $R \leftarrow \{X_{current}\}$
    $Pop \leftarrow Neighbors(H_s)$
    Evaluate(Pop)
    $archive \leftarrow nonDominatedSolutions(Pop)$
    $X_\alpha, X_\beta, X_\delta \leftarrow$ bestSolutions($archive$)
    $X_{current} \leftarrow X_\alpha$
    $R \leftarrow R \cup X_{current}$
    while $X_{current} \neq H_d$ *(ou* $X_{current} \neq B$*)* **do**
        update$_{MOGWO}(X_\alpha, X_\beta, X_\delta)$
        foreach $X_i \in Pop$ **do**
            $X_i \leftarrow$ closestHeadNeighbor(X_i)
        Evaluate(Pop)
        $NS \leftarrow nonDominatedSolutions(Pop)$
        update$_{MOGWO}(archive, NS)$
        $X_\alpha, X_\beta, X_\delta \leftarrow$ bestSolutions($archive$)
        $X_{current} \leftarrow X_\alpha$
        $R \leftarrow R \cup X_{current}$
    if $X_{current} = H_d$ **then**
        $R \leftarrow R \cup H_d$
    else
        $R \leftarrow R \cup G$

The proposed routing process is described by Algorithm 8. It is based on MOGWO and follows several steps to route a message from an object s_o to a final destination H_d or a gateway B.
Initially, the object s_o sends the message m to H_s. Once H_s receives the message, it checks if the final destination, d_o belongs to the same star (represented by S_s) as it. If it is the case, H_s can send the message directly to d_o. Otherwise, H_s must request an optimized route, R, to the final destination from the central computer OC.
To do this, OC starts by initializing a set of search agents, called Pop, with the leaders of the neighboring stars. Afterwards, it calculates the values of the objective functions of each search agent and initializes an archive with the non-dominated solutions among these agents. Then, OC selects the three best solutions in the archive. After being identified, the next leader, X_α becomes the next hop in the R route.
This process is repeated until the current leader would not be the final destination (H_d or B). At each iteration, OC updates the coordinates of the search agents based on the coordinates of X_α and those of two other leaders (X_β and $delta$), according to the equations applied by MOGWO. It also updates the archive with new non-dominated solutions and selects the next new leaders from the archive. Once the last selected leader is the final destination (H_d or B), this destination is added to route R as the last hop. This process aims to route the message in an efficient and optimized manner utilizing the intermediate leaders identified by the MOGWO algorithm.
The experiments carried out to test and evaluate the contribution described in this section and the corresponding results are presented in section 5.2 of the following chapter. This contribution is the subject of our publication [138].
We propose another routing method that relies on a geographic approach using an enhanced version of MOGWO called IMOGWO. We start by introducing IMOGWO before presenting the other routing method. Details about IMOGWO are outlined in the following section.

4.4 Improved MOGWO

The review of the state of the art in section 3.2.6, summarized in the synthesis 3.2.6, underlined the importance of making more effort to develop resolution methods that are very effective, when applied to solve standard problems and in real cases, and able to handle a large number of objectives. In this context, GWO [3] and MOGWO [2], two single-objective and multi-objective variants of the GWO meta-heuristic, were introduced and implemented to solve various engineering problems. However, the performance of MOGWO remains reduced when solving problems with a large number of objectives since it is limited to dealing with problems with only 2 or 3 objectives. Thus, our contribution described in this section consists in proposing IMOGWO, an enhanced version of MOGWO. In fact, IMOGWO adjusts the equations used in MOGWO to improve its ability to efficiently solve problems with a large number of objectives, while ensuring efficient convergence and optimal distribution of the obtained solutions. To achieve this goal, the suggested optimizer modifies the exploration method and equations employed to update the positions of the agent in MOGWO. The introduced IMOGWO is represented in Algorithm 9.

Algorithm 9: Improved Multi-objective Gray Wolf Optimizer

Input: pop, it_{max}
Output: $archive$

initialize(pop)
evaluate(pop)
$archive \leftarrow$ initializeArchive()
$x_\alpha, x_\beta, x_\delta \leftarrow$ selectLeaders($archive$)
$it \leftarrow 1$
while $it \leq it_{max}$ **do**
    $w_\alpha, w_\beta, w_\delta \leftarrow$ calculateWeights($x_\alpha, x_\beta, x_\delta,$ $bisections\_number, minValuesOfObjective, maxValuesOfObjectives$)
    foreach $x_i(t) \in pop$ **do**
        $x_{i\_gwo} \leftarrow$ getXIGWO($x_i(t), x_\alpha, x_\beta, x_\delta, w_\alpha, w_\beta, w_\delta$)
        $x_{i\_dlh} \leftarrow$ getXIDLH($x_i(t), x_{i\_gwo}$)
        $x_i(t+1) \leftarrow$ selectPosition($x_i(t), x_{i\_gwo}, x_{i\_dlh}$)
        $x_i(t) \leftarrow x_i(t+1)$
    updateArchive($archive$)
    $x_\alpha, x_\beta, x_\delta \leftarrow$ selectLeaders($archive$)
    $it \leftarrow it + 1$
return $archive$

In order to explain Algorithm 9, the latter is used, in this section, to solve a minimization optimization problem P: $min\ F(\overrightarrow{x}) = f_1(\overrightarrow{x}), f_2(\overrightarrow{x}), ..., f_o(\overrightarrow{x})$ where o is the number of objective functions $o \geq 2$. The application of IMOGWO to solve this minimization problem involves the steps described in the following subsections.

4.4.1 Initialization

The initialization phase can be divided into the steps illustrated below:

- Initialize the search space: the n agents are distributed randomly in the search space. A Pop matrix, having n rows and d columns, is filled with the coordinates of the positions of the n agents. Each agent is represented by a vector $X_i(t) = \{x_{i1}, x_{i2}, ..., x_{id}\}$.
- Evaluate the values of the objective function of each X_i in Pop.
- Initialize the archive (a storage space) with non-dominated solutions.

- Select the 3 best solutions X_α, X_β and X_δ utilizing the mechanism proposed in MOGWO. The non-dominated solutions are stored in the archive according to a grid whose dimension corresponds to the number of objectives of the optimization problem. Each solution is placed in a segment of the grid determined according to its objective values. To direct the search towards unexplored areas, the 3 best solutions are selected from the least-crowded segment (the segment of the grid containing the fewest number of solutions).

4.4.2 The different generations of IMOGWO

As the maximum number of iterations has not been reached, the calculation of the weights and the updating of the population of IMOGWO would be repeated. Each generation proceeds as follows:

4.4.2.1 Weight calculation

This step consists in calculating 3 weights: w_alpha, w_beta and w_delta. These weights are used to determine the order of importance between X_α, X_β and X_δ. MOGWO applies the same equation as GWO (Algorithm 1) to calculate the positions of the wolves, ensuring that each element follows the 3 best solutions (X_α, X_β and X_δ). These solutions have the same importance and a coefficient equal to 1. However, they are classified according to their performance. The coefficients w_alpha, w_beta and w_delta are, then, employed to weight the respective contributions of X_α, X_β and X_δ. Thus, the equation utilized to update the coordinates of the search agents is expressed as follows (Equation 4.10):

$$\overrightarrow{X_i}(t+1) = (w\_alpha * \overrightarrow{X_{i1}} + w\_beta * \overrightarrow{X_{i2}} + w\_delta * \overrightarrow{X_{i3}}) \tag{4.10}$$

The values of w_alpha, w_beta and w_delta are determined using the following method. The classification of the three best solutions is considered as a separate optimization problem denoted $P_{ranking}$. This problem shares the same objectives as the initial problem P: $f_1(\overrightarrow{x})$, $f_2(\overrightarrow{x})$, ..., $f_o(\overrightarrow{x})$. Since $P_{ranking}$ consists only of solutions X_α, X_β and X_δ, its dimension is set to 3. Therefore, exact solution methods, used to solve small-scale optimization problems, can be applied to solve $P_{ranking}$. We use the weighting technique, which is the most direct exact method applied to solve low-dimensional multi-objective problems. It consists in transforming the initial multi-objective problem into a single-objective problem by assigning an order of priority and associating a weighting coefficient to each objective. Then, the weighted objectives are summed to obtain a new objective function. Thus, the initial problem is transformed into a single-objective problem aiming to maximize $\sum_{i\in 1..o} \lambda_i * f_i(x)$ over $X_{ranking}$ where λ_i represents the weighting coefficient associated

Figure 4.3: Weight calculation for X_α, X_β and X_δ

	x_alpha obj_1_val	...	x_alpha obj_o_val	x_beta obj_1_val	...	x_beta obj_o_val	x_delta obj_1_val	...	x_delta obj_o_val
x_alpha obj_1_val	0	...	0	score_1_a_b	...	0	score_1_a_d	...	0
...	0	...	0	...	...	...	...	...	...
x_alpha obj_o_val	0	...	0	0	...	score_o_a_b	0	...	score_o_a_d
x_beta obj_1_val	score_1_b_a	...	0	0	...	0	score_1_b_d	...	0
...	...	...		0	...	0	...	...	...
x_beta obj_o_val	0	...	score_o_b_a	0	...	0	0	...	score_o_b_d
x_delta obj_1_val	score_1_d_a	...	0	score_1_d_b	...	0	0	...	0
...	...	...	...	...	...	...	0	...	0
x_delta obj_o_val	0	...	score_o_d_a	0	...	score_1_d_b	0	...	0

	score_sum_i	Weight_i
	score_1_a_b +...+ score_1_a_d + ... + score_o_a_b + ... + score_o_a_d	100* score_sum_a/ total_sum
	score_1_b_a +...+ score_1_b_d + ... + score_o_b_a + ... + score_o_b_d	100* score_sum_b/ total_sum
	score_1_d_a +...+ score_1_d_b +...+ score_o_d_a +...+ score_1_d_b	100* score_sum_d/ total_sum
total_sum	$\sum$ score_sum_i	

with the objective function f_i. This transformation simplifies the ranking of the possible solutions because the different values of the single objective can be directly ordered.

The values of w_alpha, w_beta and w_delta are obtained by applying Algorithm 10, which takes as input the solutions X_α, X_β and X_δ and returns their respective weights.

Algorithm 10 uses the cross-sort technique [139] to apply the exact weighting method. First, a square matrix of dimensions $(3 * o) * (3 * o)$ is filled with integer values assigned to the objective functions of each solution. Each row and column of this matrix corresponds to an objective function of one of the solutions among X_α, X_β and X_δ. The scores to be entered in each case are calculated from the values of the objective functions associated with the row and column of the box. These scores are computed employing Algorithm 11 having, as input, the values of the objective functions. First, Algorithm 11 divides the range of values of the objective function into a number of bisections denoted $bisection\_number$. Thus, each sub-interval obtained has a width d_value. The score returned by this algorithm corresponds to the difference between the values of two objective functions. This difference is divided by d_value. To avoid a zero score, it is incremented by 1 in all cases, except in special cases where the corresponding solutions are identical or when the case belongs to both a row and a column of two different objective functions. Secondly, the scores of each solution among X_α, X_β and X_δ are summed. Then, the total sum of all scores, $total\_sum$, is calculated. The weight of each solution corresponds to the percentage of the sum of its scores compared to $total\_sum$. Figure 4.3 illustrates the general structure of the utilized matrix and the procedure followed to calculate the weights.

Algorithm 10: calculateWeights() algorithm

Input: x_alpha, x_beta, x_delta, obj_number, $bisection\_number$, $minValuesOfObjectives$, $maxValuesOfObjectives$

Output: w_alpha, w_beta, w_delta

$m \leftarrow$ a square matrix of dimension $(3 * obj\_number) * (3 * obj\_number)$
foreach $i \in 3 * obj\_number$ **do**
    foreach $j \in 3 * obj\_number$ **do**
        if i *Div* $obj\_number = 0$ **then**
            $sol_1 \leftarrow x\_alpha$
        else
            if i *Div* $obj\_number = 1$ **then**
                $sol_1 \leftarrow x\_beta$
            else
                $sol_1 \leftarrow x\_delta$
        if j *Div* $obj\_number = 0$ **then**
            $sol_2 \leftarrow x\_alpha$
        else
            if j *Div* $obj\_number = 1$ **then**
                $sol_2 \leftarrow x\_beta$
            else
                $sol_2 \leftarrow x\_delta$
        $objective\_index_1 \leftarrow i$ Mod obj_number
        if $objective\_index_1 = 0$ **then**
            $objective\_index_1 \leftarrow obj\_number$
        $objective\_index_2 \leftarrow j$ Mod obj_number
        if $objective\_index_2 = 0$ **then**
            $objective\_index_2 \leftarrow obj\_number$
        $m[i][j] \leftarrow$ getValue($sol_1.obj\_value$, $sol_2.obj\_value$, $objective\_index_1$, $objective\_index_2$, obj_number, $bisection\_number$, $minValuesOfObjectives[objective\_index_1]$, $maxValuesOfObjectives[objective\_index_2]$)
$score\_sum\_alpha \leftarrow \sum_{r \in 1..obj\_number} \sum_{c \in obj\_number+1..3*obj\_number} m[r][c]$
$score\_sum\_beta \leftarrow \sum_{r \in obj\_number+1..2*obj\_number} \sum_{c \in 1..obj\_number, 2*obj\_number..3*obj\_number} m[r][c]$
$score\_sum\_delta \leftarrow \sum_{r \in 2*obj\_number+1..3*obj\_number} \sum_{c \in 1..2*obj\_number} m[r][c]$
$total\_sum \leftarrow score\_sum\_alpha + score\_sum\_beta + score\_sum\_delta$
$w\_alpha \leftarrow 100 * score\_sum\_alpha / total\_sum$
$w\_beta \leftarrow 100 * score\_sum\_beta / total\_sum$
$w\_delta \leftarrow 100 * score\_sum\_delta / total\_sum$
return w_alpha, w_beta, w_delta

Algorithm 11: getValue() algorithm

Input: $sol_1.obj\_value$, $sol_2.obj\_value$, $objective\_index_1$, $objective\_index_2$, o, $bisection\_number$, $minValuesOfObjective$, $maxValuesOfObjective$

Output: $score$

if $sol_1 = sol_2$ **then**
  $score \leftarrow 0$
else
  if $objective\_index\_1 <> objective\_index_2$ **then**
    $score \leftarrow 0$
  else
    $d\_value \leftarrow (maxValuesOfObjective - minValuesOfObjective)/bisection\_number$
    $score \leftarrow ((sol\_1.obj\_value - sol\_2.obj\_value) \text{ Div } d\_value) + 1$
return $score$

4.4.2.2 Update of the IMOGWO population

For each $X_i(t)$ in Pop, the new position, $X_i(t+1)$, is calculated as follows:

- Use the values of w_alpha, w_beta and w_delta to determine the first candidate position $X_{i-GWO}(t+1)$ applying Equation 4.10.
- Computing a second candidate position $X_{i-DLH}(t+1)$. We combine the calculation of a second position, utilized in I-GWO, with the multi-objective algorithm. The method described in paragraph (**a**) (subsection 3.2.4.2) is used to determine the second candidate position of each wolf at each iteration.
- Select $X_i(t+1)$ from $X_i(t)$, $X_{i-GWO}(t+1)$ and $X_{i-DLH}(t+1)$ by applying Algorithm 12. Details about the coordinates of the research agent are updated only if this update optimizes the objectives. First, Algorithm 12 chooses either X_{i-GWO} or X_{i-DLH} by selecting the solution that dominates the other. If neither solution dominates the other, X_{i-GWO} is chosen. Afterwards, it chooses either the previously-chosen solution X_j or the initial solution $X_i(t)$. The algorithm selects the solution that dominates the other. Otherwise, it returns $X_i(t)$ and, therefore, keeps the position of the search agent unchanged.

4.4.2.3 Update of the archive

This step consists in updating the archive with the new non-dominated solutions, in accordance with the process applied by MOGWO. In other words, each non-dominated found solution is added to the archive following a process that includes checking the domination of that solution by those already present in the archive and removing the dominated solutions, if necessary. If no solution in the archive dominates the new solution, it is added to the archive. Otherwise, it is rejected.

Algorithm 12: selectPosition() algorithm

Input: $X_i(t)$, X_{i-GWO}, X_{i-DLH}
Output: $X_i(t+1)$

if $X_{i-GWO} \succ X_{i-DLH}$ **then**
  $X_j \leftarrow X_{i-GWO}$
else
  if $X_{i-DLH} \succ X_{i-GWO}$ **then**
    $X_j \leftarrow X_{i-DLH}$
  else
    $X_j \leftarrow X_{i-GWO}$
if $X_j \succ X_i(t)$ **then**
  $X_i(t+1) \leftarrow X_j$
else
  if $X_i(t) \succ X_j$ **then**
    $X_i(t+1) \leftarrow X_i(t)$
  else
    $X_i(t+1) \leftarrow X_j$
return X_j

4.4.2.4 The selection of the best solutions

In this step, the 3 new best solutions (X_α, X_β and X_δ) are selected from the solutions of the least-crowded segment in the archive.

4.4.3 The final solutions

The items that remain in the archive after the end of all IMOGWO generations are the final solutions to be returned.
The performance of the proposed IMOGWO was evaluated by applying it to solve DTLZ benchmark problems and real-world multi-objective geographic routing modeling problem in an IoT network. The conducted experiments and the obtained results are presented in the corresponding sections of the following chapter: section 5.3 for the resolution of DTLZ problems and section 5.5 for the multi-objective geographic routing problem.
Since geographic routing requires determining the positions of the objects, a localization solution becomes necessary in order to develop it. For this reason, we introduce an indoor location solution in IoT networks. We benefit from Deep Learning techniques to develop an effective approach for indoor localization. Therefore, our contribution, detailed in the following section, focuses on Deep Learning techniques.

4.5 Optimization of DL parameters using metaheuristics

The literature review presented in section 3.3.6, summarized in the synthesis 3.3.6, highlighted the importance of developing the optimization of the weights of Deep Learning models using metaheuristics. This approach makes it possible to accelerate the convergence of DL models towards the optimal weight values and improve their performance. Furthermore, the developed optimizers should be designed in such a way that it can be applied to different DL techniques rather than being specific to a single method. Our contribution, presented in this section, falls into this context. It consists in modeling the training of a DL model in the form of a single-objective optimization problem by establishing a link between the adjustment of the parameters of the layers of neurons and solving an optimization problem employing a metaheuristic algorithm. The objective of this approach is to develop an optimizer used to train DL models by hybridizing these two beneficial paradigms (optimization and DL), while being generalizable and applicable to several DL techniques.

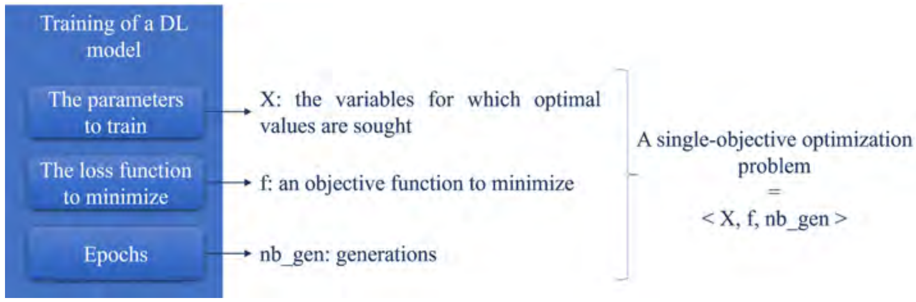
In the suggested contribution, a Deep Learning model comprising dense layers is utilized and the learning of their weights is optimized using metaheuristic algorithms. As mentioned in subsection 3.3.3, during the training phase, many learning iterations are carried out in order to determine the values of the weights making it possible to get closer to the correct function that maps the inputs x to outputs y. In other words, these weights are learned and improved during training. Equation 3.19 simplifies and generalizes the output equation of a layer used in a neural network.

The weights are adjusted, at each iteration, using an optimization algorithm to minimize prediction error during epochs. The effectiveness of the optimizer applied during training plays a crucial role in increasing the speed of the DL model convergence and improving the quality of its prediction. In the IoT applications, where increased accuracy and fast processing are required, it is important to enhance the performance of the location model by utilizing more efficient optimizer. To overcome this challenge, we use optimization metaheuristics. By treating the learning of the DL model weights as a metaheuristic optimization problem, we can apply specific algorithms to generate solutions and update these weights iteratively.

Based on the proposed design of the learning stage, the weights of a DL model are optimized, in this study, utilizing metaheuristic algorithms. Weight learning is, thus, formulated as a single-objective optimization problem. Figure 4.4 illustrates the transition from training a DL model to resolving a single-objective optimization problem. This transition involves redefining the components and objectives of a DL model training in an optimization problem having single objective to minimize. The components are transformed as follows:

- The parameters to be trained are variables for which optimal values are searched. In DL, these parameters designate the weights of the neural network. These weights are adjusted

Figure 4.4: Transition from training a DL model to resolving a single-objective optimization problem



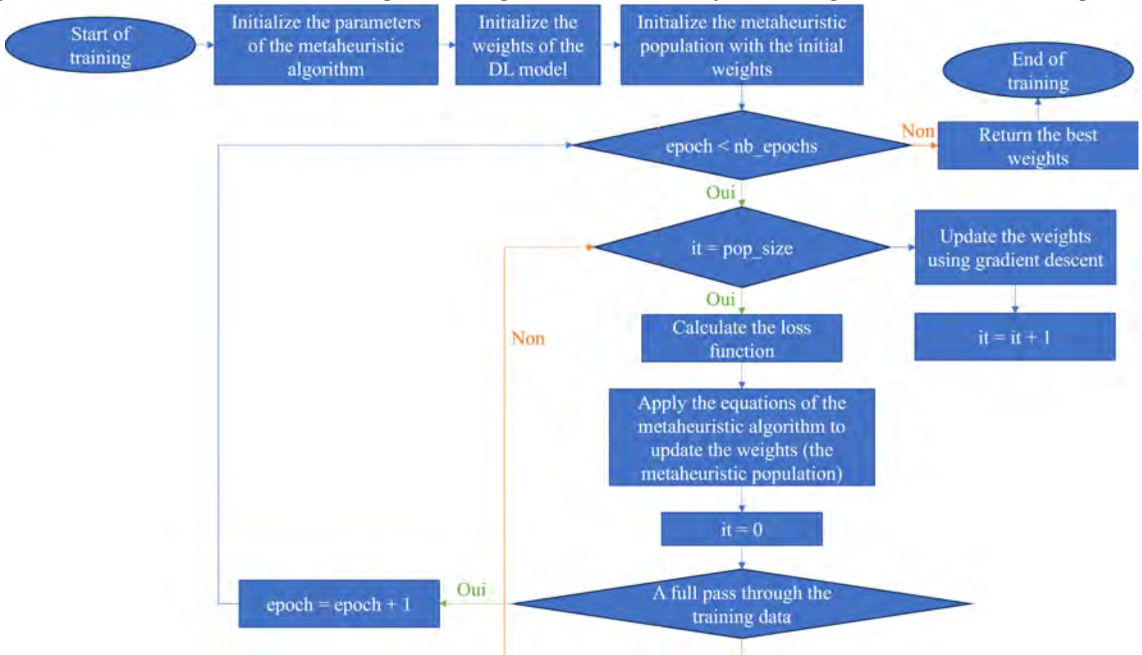
during training to minimize the loss function. In the context of single-objective optimization, these parameters become the decision variables X used to optimizing the objective function.

- The loss function (prediction error) to be minimized becomes the objective function. In DL, this function is applied to measure the error between the predicted and actual values of variables. In the single-objective optimization problem, the objective function f is defined to be minimized by adjusting the decision variables X representing the parameters of the DL model.

- Epochs are assimilated to generations. In DL, the epochs correspond to the number of times in which the entire training data is processed by the model. In single-objective optimization, the number of generations represents the iterations or generations of the optimization algorithm. In each iteration, the algorithm evaluates the objective function and adjusts the decision variables to obtain better solutions, as it is the case of deep training performed over multiple epochs to improve the model performance.

The main idea of the approach is to consider the DL training process as an optimization problem whose objective is to minimize the loss by adjusting the DL model parameters (decision variables) through a series of iterations or generations. In this approach, different single-objective metaheuristic optimization techniques can be utilized to refine the model and enhance its performance.
The representation of weight optimization using a metaheuristic method is illustrated in Figure 4.5 and detailed in Algorithms 13, 14 and 15.
At the start of the training phase, the weights are initialized and a maximum number of epochs is set. These initial weights are employed to begin the population of the metaheuristic optimization algorithm. Then, a new iteration starts until reaching the maximum number of epochs. At each epoch, the entire training data set is used and divided into batches to be processed. At the end of each batch, the value of the loss function is calculated. It is, subsequently, checked whether the population of the metaheuristic optimization was completely constructed or updated with the latest

Figure 4.5: Flowchart of training the weights of neuron layers using a metaheuristic algorithm



iterations. In this case, the mathematical equations employed in metaheuristic optimization are applied to compute new weights and the population of the algorithm is updated according to these new weights. Otherwise, the weights are updated utilizing the gradient descent equations. Once the maximum number of epochs was exceeded, the weights obtained during the last iteration would be considered optimal, which shows the end of the training phase.

Algorithm 13 describes in details the training phase of the developed model. The model must be trained using a computer given the training data as input in order to provide a trained model that can produce accurate results in the prediction phase. First, the weights of the neural network and the parameters of the metaheuristic algorithm to apply are initialized with random values. As instances of these parameters, we can cite the notably-specified size of the population to be considered. It is denoted pop_size. Then, a number of epochs are executed to train the model. Each epoch includes a number of iterations equal to the size of the input data divided by the size of the batch to be considered.

Algorithms 14 and 15 depict the process performed at each iteration. During different epochs, the metaheuristic is applied after each pop_size iteration. In other words, the training phase is divided into steps having length pop_size iterations. At the first step, the weight values are used to initialize the population. At the end of this step, the metaheuristic algorithm is utilized to update the weights. In the subsequent steps, the weights generated by the previous step are initially employed in order to make predictions. At the end of each step, the weights are optimized using the equations of the applied metaheuristic algorithm.

Algorithm 13: Training the weights using a metaheuristic algorithm

Input: *input_data*, *pop_size*, *nb_generations*, *nb_epochs*
Output: *parameters*

```
weights ← random weights
metaheuristic_parameters ← initializeMetaheuristicParameters()
it ← 0
c_pop_write ← 0
c_pop_read ← 1
apply_metaheuristic_algo ← False
first_pop ← True
pop ← ∅
foreach epoch ∈ nb_epochs do
    foreach step ∈ nb_iterations do
        if apply_metaheuristic_algo est False then
            constructOrUsePop(input_data,weights,pop, first_pop, pop_size, c_pop_write,
              c_pop_read, apply_metaheuristic_algo)
        else
            weights ← trainWithMetaheuristicAlgo(pop, nb_generations, it,
              metaheuristic_parameters)
```

To conclude, at each iteration, three scenarios are possible: (i) the DL parameters are initialized randomly and added to the initial population of the optimization algorithm ; (ii) the elements of the population are updated using the equations of the metaheuristic algorithm ; (iii) an element of the population already modified by the optimization algorithm is selected to be utilized during the training phase.

Algorithm 14 handles situations (i) and (iii), while Algorithm 15 deals with case (ii). More precisely, Algorithm 14 is used in iterations where the metaheuristic optimization algorithm has not been yet applied. In the first case, during the initial construction of the population, the training parameters are randomly generated by the classical method and added to the population. Afterwards, the size of the built population is checked. If it reaches its maximum size, two actions are made:

- The boolean variable $first\_pop$ is toggled to $False$ in order to prevent the repetition of this process because the population is randomly-initialized only once.
- The boolean variable $apply\_metaheuristic\_algo$ is activated, indicating that the population will be updated by applying the metaheuristic algorithm in the next iteration.

Algorithm 14 handles scenario (iii) where both $apply\_metaheuristic\_algo$ and $first\_pop$ are set to $False$. In this situation, an element of the population is selected to be used during the training of the DL model. Subsequently, the counter c_pop_read, utilized to iterate through the elements in the population, is incremented to move to the next element in the next iteration. If c_pop_read exceeds the population size, this means that all elements in the population were used and must be updated by the metaheuristic algorithm in the next iteration. Therefore, $apply\_metaheuristic\_algo$ is modified into $True$.

Algorithm 15 describes scenario (ii) where the equations used in the metaheuristic optimization algorithm are applied to adjust the population elements (the parameters) for subsequent iterations. After this adjustment, the first element of the population will be extracted to be used as parameters of the DL model in the next iteration. This procedure ensures that the population evolves and adapts to improve the optimization process. The parameters of the optimization algorithm are also updated employing the corresponding equations in order to be utilized in the next generation.

Algorithm 14: constructOrUsePop() algorithm

Input: pop, $first\_pop$, pop_size, c_pop_write, c_pop_read, $apply\_metaheuristic\_algo$
Output: $parameters$

if $first\_pop$ *is* $True$ **then**
    $parameters \leftarrow$ getParametersWithTraditionalMethod()
    $pop \leftarrow pop \cup parameters$
    $c\_pop\_write \leftarrow c\_pop\_write + 1$
    if $c\_pop\_write = pop\_size$ **then**
        $first\_pop \leftarrow$ False
        $apply\_metaheuristic\_algo \leftarrow$ True
else
    $parameters \leftarrow pop[c\_pop\_read]$
    $c\_pop\_read \leftarrow c\_pop\_read + 1$
    if $c\_pop\_read > pop\_size$ **then**
        $apply\_metaheuristic\_algo \leftarrow$ True

Algorithm 15: trainWithMetaheuristicAlgo() algorithm

Input: pop, $nb\_generations$, $apply\_metaheuristic\_algo$, $metaheuristic\_parameters$
Output: new pop

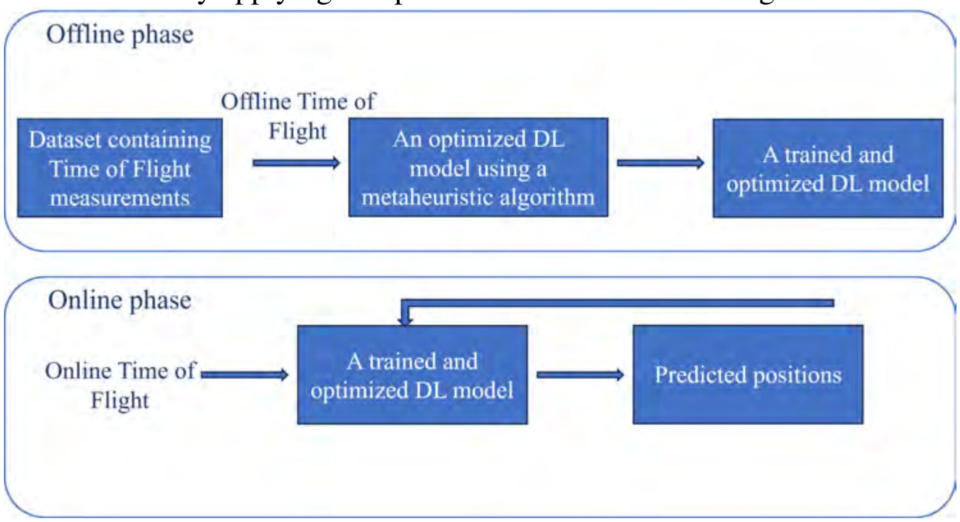
updatePopWithMetaheuristicAlgo(pop, $metaheuristic\_parameters$)
$it \leftarrow it + 1$
$metaheuristic\_parameters \leftarrow$ updateParametersOfMetaheuristicAlgo($nb\_generations$, it)
$apply\_metaheuristic\_algo \leftarrow False$
$parameters \leftarrow pop[1]$
$c\_pop\_read \leftarrow 2$
return pop

The approach presented in this section was implemented to design an indoor localization solution in IoT networks. It is tested and evaluated in the 5.4 section of the next chapter. This localization solution is based on the combination of Deep Learning techniques and metaheuristic methods. The following section will detail this approach.

4.6 Indoor localization by hybridizing DL techniques and meta-heuristics

This section defines our contribution to indoor localization in an IoT network. The proposed approach is based on the previously-presented contribution as it relies on a Deep Learning model optimized and trained using a metaheuristic optimization algorithm. The localization process to be developed includes two stages (Figure 4.6): an offline phase and an online phase.

Figure 4.6: Localization by applying an optimized DL model and using UWB ToF measurements

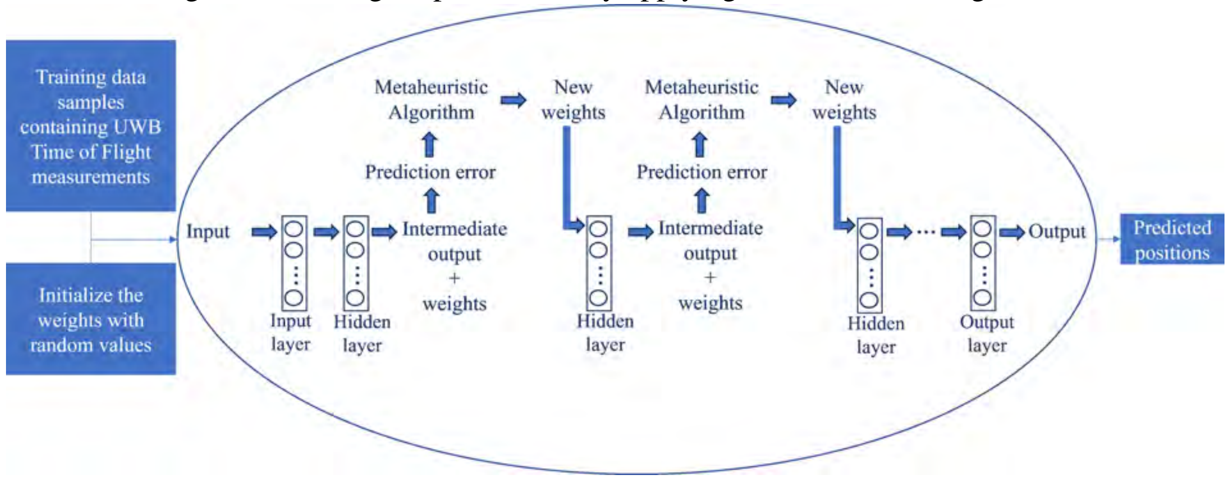


4.6.1 Offline phase

In this first phase, called the offline phase (Figure 4.6(a)), we train the localization model using the previously-prepared dataset comprising mainly the measurements of UWB time of flight. These measurements are taken from UWB signals produced by communications between the connected objects whose positions are known and the fixed anchors, leveraging the high precision of Time-of-Flight (ToF) measurements enabled by UWB technology. The prepared data serve as the input of the DL model to be trained. In this context, the real LocURa4IoT platform [140] offers a dataset specially designed to develop and evaluate indoor localization methods in IoT environments, based on ToF measurements and UWB technology. This dataset is available online [141].

The main objective of this phase is to make the model learn the association of UWB time-of-flight measurements to the coordinates of the mobile object, so that it can accurately predict the position of the object based on these measurements. In other words, the model learns the function to map the measurements of the ToF towards the corresponding coordinates of an object. To do this, a metaheuristic optimization algorithm is utilized to train a DL model on the input data. Figure 4.7

Figure 4.7: Weight optimization by applying a metaheuristic algorithm



illustrates the training process. During the latter, the model adjusts its internal weights to minimize a specific loss function that measures the difference between the positions predicted by the model and the actual positions of the mobile object. This adjustment is done utilizing a metaheuristic algorithm as it guides the optimization process making the model to improve over time by reducing the difference between its predictions and the true positions.

In addition, to evaluate the performance of the proposed model during training and avoid overfitting, part of the data is used for validation phase, which allows monitoring the ability of the model to generalize its predictions to new, previously-unobserved data. At the end of this phase, a trained and optimized localization model, to be used in the next phase of the localization process, is obtained.

4.6.2 Online phase

The online phase (Figure 4.6(b)) is an impotant step of the application of the introduced approach of indoor localization in an IoT network. We consider a scenario where a mobile object, equipped with UWB technology, moves in the same space used in the offline phase. Fixed anchors emit UWB signals, allowing the mobile object to determine its position by measuring the flight times of these signals between the anchors and the mobile object. When the latter wants to locate itself, it begins by exchanging messages with the fixed anchors using, for example, a TWR [38] protocol or an improved version of it. This exchange allows the object to obtain UWB ToF measurements. Once the mobile object has collected the necessary data, it transmits them to the DL localization model previously trained and optimized during the offline phase. This model uses UWB ToF measurements as input to estimate the position of the mobile object in the space covered by the anchors.

In summary, in the suggested approach, a DL-based localization model, trained using a metaheuristic optimization algorithm, is used to estimate the position of a mobile object based on the UWB ToF measurements collected during messages exchanges with fixed anchors. This approach allows the mobile object to localize itself in an indoor environment by benefiting from the advantages of UWB technology and synergistically combining the two important paradigms: metaheuristic optimization and DL techniques.
The experiments conducted to evaluate the indoor localization contribution described in this section and the corresponding results are presented in5.4 section of the following chapter. This contribution is the subject of our publication [142].
The approaches described in the previous three sections are the basics of building a multi-objective geographic routing solution in an IoT network presented in detail in the next section.

4.7 A geographic routing based on IMOGWO

This section focuses on our contribution which consists in a hybrid and multi-objective geographic routing approach to be applied in IoT networks having mesh architecture. Geographic routing requires prior knowledge about the positions of the connected objects in order to effectively direct messages through the network. To do this, the developed routing method uses the localization approach previously described in section 4.6, ensuring precise knowledge of the positions of objects. Furthermore, the proposed routing solution is multi-objective as it applies the suggested IMOGWO approach, presented in section 4.4, to optimize the routing process by taking into account the objectives described in section 4.2.2 simultaneously. These objectives include positive and efficient progress towards the final destination, optimal energy management, overall reliability in communications, hop continuity to the final destination and the optimization of transmission latency. The goal of the approach is to ensure efficient message routing while meeting the requirements of IoT networks. In our approach, the construction of routes is distributed among the objects of the IoT network without requiring a central computer to manage routing. This decentralization aligns with the fundamental idea of Edge Computing. In fact, this idea consists of processing data and performing decision-making as close as possible to the data sources in order to reduce latency, save bandwidth, and enhance system resilience. This section presents the considered network model to consider as well as the assumptions to take into account. In addition, it describes the algorithms showing the routing process to be developed.

4.7.1 The network model

The proposed IoT network model is characterized by a mesh topology where each node is connected directly to all other nodes within its communication range. The utilized communication technology is called UWB. This network includes various objects (stationary and mobile objects) having different energy capacities. To ensure precise localization of mobile objects, fixed anchors are deployed in the network. The use of mesh topology IoT network may lead to the appearance of some challenges. In fact, environmental dynamics can result in unanticipated changes in the network topology, making it difficult to select optimal paths for data transmission.

4.7.2 Assumptions

The assumptions to be considered during the routing process are listed below:

- The network is composed of heterogeneous objects (stationary objects and mobile objects) exhibiting different energy capacities.
- Routing is multi-hop ; i.e. , the construction of a route is distributed and shared between the connected objects ; each of which calculates the next hop.
- The objective functions of routing are evaluated at each hop.
- If an object faces some problems during routing (e.x., it is unable to move forwards), it is necessary to go back (by one hop) and start again from the previous object in the traveled route, resuming positive progress as soon as possible.
- Objects start with the same degree of confidence which changes over time during previous experiences with known (neighboring) objects.
- To route messages, each object must know its own position in the network, the location of the final destination and information about its neighbors at two hops (position, residual energy, confidence level).
- Each mobile object transmits its position to its neighbors as soon as it moves in a specified reference distance.
- Each object regularly transmits information to its neighbors, including its position, confidence levels, residual energy and the number of neighbors of known objects.
- Each object listens to its neighbors to build and update a 1- and 2-hop neighboring table including position, residual energy and the confidence level. Figure 4.8 presents an explanatory example of a neighboring table of an object in the network.

Figure 4.8: An explanatory example of a neighboring table for an object in the network

Neighboring table for Object 20

id	x	y	Energy	Confidence	Neighbors				
					id	x	y	Energy	Confidence
10	22.1	27.7	21403.0	0.72	39	30.45	24.4	18010.0	0.82
					46	24.6	33.16	13330.0	0.66
					73	17.3	24.6	12720.0	0.43
					...				
32	10.64	2.29	12102.0	0.3	37	6.24	1.33	19246.0	0.94
					12	12.46	3.31	17303.0	0.73
					...				
...									

4.7.3 The routing process

The routing proposed is geographic, reactive and multi-objective. It is essentially based on the positions of the connected objects in the network to route messages. The introduced routing approach considers the objectives described in section 4.2.2 in order to optimize message routing and guarantee efficient communications. Hybrid routing involves maintaining local knowledge of the topology at each object, up to a distance of 2 hops by exchanging periodically control messages. Thus, each object constructs a neighboring table containing information about its neighbors accessible at 1 and 2 hops. The built table is updated by the received control messages. Therefore, the routes towards 1- and 2-hop neighbors are known pro-actively, while the full, end-to-end path is discovered reactively.

The routing process starts with Algorithm 16. It begins by checking the presence of the final destination object among the neighbors accessible at 1 or 2 hops in the neighboring table 2 hops away from the initial source. If so, the routing is proactive, meaning the route is already known. In this case, the source object sends the message directly to the final destination object or through a neighboring object. On the other hand, if the final destination object is not listed in the neighboring table, reactive geographic routing is activated (Algorithm 17) in order to determine the best route to transmit the message.

Algorithm 17 is applied to determine an optimized route to transmit the message m from the initial source object s_o to the final destination object d_o by applying the IMOGWO algorithm. The process begins by determining the position of the final destination object d_o by sending queries in the network from the neighbors of the current object. If the source object s_o is mobile, its position is calculated using the Deep Learning localization model, $DLModel$, after exchanging messages with the fixed anchors used to localize mobile objects. Afterwards, the algorithm initializes the population of IMOGWO Pop, by including the neighbors of the current object s_o. Then, it evaluates

Algorithm 16: IMOGWO-based routing algorithm

```
Input: m, s_o, d_o, DLModel
Output: route m to d_o
if d_o ∈ N_T(s_o) then
    if d_o ∈ N(s_o) then
        s_o transmits m to d_o
    if d_o ∈ N2H(s_o) then
        s_o transmits m to o_v(∈ N(s_o))
        o_v transmits m to d_o
else
    R ← lookForPathToDestination(s_o, d_o, m, DLModel)
```

the members of the population by computing their fitness for each objective function and utilizing the positions of s_o and d_o.

After identifying the non-dominated solutions NDS, the algorithm initializes an archive and selects the best leaders among these solutions by applying the mechanisms of the IMOGWO algorithm. Subsequently, the source object s_o sends the message m and information about the selected leaders and the population Pop to X_α, the best among the chosen leaders. Therefore, X_α becomes the current object c_o (where the message arrives during routing). Then c_o is iteratively updated until it reaches the final destination object d_o. At each iteration, if the current object c_o is mobile, its position is computed using the location model $DLModel$. Subsequently, the coordinates of the members of the population Pop are updated using the equations applied by the IMOGWO algorithm (Algorithm 18). Then, for each element X_i of the population Pop, a matching method is used to match the calculated coordinates of X_i with the coordinates of a real object among $N(c_o)$, the set of neighbors of the current object c_o (Algorithm 21). Later, for each element of the population, the fitness of each objective function is computed. After these calculations, the algorithm searches the non-dominated solutions from the elements of the population Pop and updates the archive with these solutions employing the IMOGWO algorithm (Algorithm 22). Finally, the new best solutions are selected from the elements of the archive utilizing the mechanisms of the IMOGWO algorithm and the selected leaders are excluded from the archive during the next iteration.

Finally, the current object c_o transmits the message to the next leader X_α and decides to follow the route determined by the algorithm or to return back according to the availability of the neighbors allowing positive progress towards the final destination object. The reaching of d_o by c_o means that the final destination was attained and the routing of the message is accomplished.

Algorithm 18 allows updating the coordinates of the object X_i applying the equations used by the IMOGWO algorithm. It takes as input the object X_i as well as the leaders X_α, X_β et X_δ. It starts by computing X_{i-GWO} using GWO with X_i, X_α, X_β and X_δ as input. Then it calculates X_{i-DLH} using DLH having X_{i-GWO} as input. The calculation of these two positions is presented

Algorithm 17: lookForPathToDestination() algorithm

Input: $s_o, d_o, m, DLModel$
Output: The route from s_o to d_o

$(d_x, d_y) \leftarrow$ searchPosition(d_o)
if *isMobile(s_o)* **then**
    $(s_x, s_y) \leftarrow$ computePosition(s_o, $DLModel$)
$c_o \leftarrow s_o$
$Pop \leftarrow \varnothing$
foreach $v_i \in N(c_o)$ **do**
    $Pop \leftarrow Pop \cup \{v_i\}$
foreach $X_i \in Pop$ **do**
    foreach $f_i \in$ *objectiveFunctions* **do**
        Calculate the fitness of X_i for f_i
$NDS \leftarrow$ findNonDominatedSolutions(Pop)
$archive \leftarrow$ initializeArchive(NDS)
$\{X_\alpha, X_\beta, X_\delta\} \leftarrow$ selectLeaders($archive$)
exclude $\{X_\alpha, X_\beta, X_\delta\}$ from $archive$
s_o transmits m, X_β, X_δ and Pop to X_α
$c_o \leftarrow X_\alpha$
while $c_o \neq d_o$ **do**
    if *isMobile(c_o)* **then**
        $(c_x, c_y) \leftarrow$ computePosition(c_o, $DLModel$)
    foreach $X_i \in Pop$ **do**
        updateCoordinates($X_i, X_\alpha, X_\beta, X_\delta$)
    foreach $X_i \in Pop$ **do**
        $X_i \leftarrow$ mapFromComputedCoordinatesToRealPosition(c_o,X_i,$N(c_o)$)
    foreach $X_i \in Pop$ **do**
        foreach $f_i \in$ *objectiveFunctions* **do**
            Calculate the fitness of X_i for f_i
    $NDS \leftarrow$ findNonDominatedSolutions(Pop)
    updateArchive($archive$, NDS)
    $\{X_\alpha, X_\beta, X_\delta\} \leftarrow$ selectLeaders($archive$)
    exclude $\{X_\alpha, X_\beta, X_\delta\}$ from $archive$
    c_o transmits m, X_β, X_δ and Pop to X_α
    if *canEnablePositiveProgress($N(c_o)$)* **then**
        $c_o \leftarrow X_\alpha$
    else
        $c_o \leftarrow$ previous c_o

in Algorithms 19 and 20. Then, the algorithm compares solutions X_{i-GWO} and X_{i-DLH} to determine which one dominates the other. If X_{i-GWO} dominates X_{i-DLH}, then $X_{i-(t+1)}$ is updated by X_{i-GWO}. If X_{i-DLH} dominates X_{i-GWO}, then $X_{i-(t+1)}$ is updated with X_{i-DLH}. If neither position dominates the other, then $X_{i-(t+1)}$ is chosen randomly between X_{i-GWO} and X_{i-DLH}. Finally, if solution $X_{i-(t+1)}$ dominates the current solution X_i, the coordinates of X_i are updated with those of $X_{i-(t+1)}$.

Algorithm 18: updateCoordinates() algorithm

Input: $X_i, X_\alpha, X_\beta, X_\delta$
Output: update X_i

$X_{i-GWO} \leftarrow$ calculateXGWO($X_i, X_\alpha, X_\beta, X_\delta$)
$X_{i-DLH} \leftarrow$ calculateXDLH(X_{i-GWO})
if $X_{i-GWO} \succ X_{i-DLH}$ **then**
  $X_{i-(t+1)} \leftarrow X_{i-GWO}$
else
  if $X_{i-DLH} \succ X_{i-GWO}$ **then**
    $X_{i-(t+1)} \leftarrow X_{i-DLH}$
  else
    $X_{i-(t+1)} \leftarrow rand(X_{i-GWO}, X_{i-DLH})$
if $X_{i-(t+1)} \succ X_i$ **then**
  $X_i \leftarrow X_{i-(t+1)}$

Algorithms 19 and 20 model the calculation of the new candidate's positions of X_i applying the equations written in section 4.4.

Algorithm 19: calculateXGWO() algorithm

Input: $X_i, X_\alpha, X_\beta, X_\delta$
Output: X_{i-GWO}

$D_\alpha \leftarrow |C_1.X_\alpha - X_i|$
$D_\beta \leftarrow |C_2.X_\beta - X_i|$
$D_\delta \leftarrow |C_3.X_\delta - X_i|$
$X_1 \leftarrow X_\alpha - A_1.(D_\alpha)$
$X_2 \leftarrow X_\beta - A_2.(D_\beta)$
$X_3 \leftarrow X_\delta - A_3.(D_\delta)$
$X_{i-GWO} \leftarrow (X_1 + X_2 + X_3)/3$
return X_{i-GWO}

Algorithm 21 is utilized to map the coordinates of an object X_i, from the elements of Pop, towards the coordinates of a real object among the neighbors of the current object c_o. The algorithm starts by initializing an empty list $positions$ to store the positions of the current object's neighbors. Then, the algorithm adds the position of each neighbor v_j in the set $N(c_o)$ to the list $positions$. After collecting the positions of all neighbors, the algorithm determines the nearest neighbor $V_{nearest}$

Algorithm 20: calculateXDLH() algorithm

Input: X_{i-GWO}, Pop
Output: X_{i-DLH}

$R_i \leftarrow ||X_i - X_{i-GWO}||$
$N_i \leftarrow \{X_j | euclidean\_distance(X_i, X_j) \leq R_i, X_j \in Pop\}$
foreach $d \in \{1, 2, ..., D\}$ **do**
  $X_n \leftarrow rand(N_i)$
  $X_r \leftarrow rand(Pop)$
  $X_{i-DLH,d} \leftarrow X_{i,d} + (X_{n,d} - X_{r,d})$
return X_{i-DLH}

based on the recorded positions. Finally, it returns the position of the nearest neighbor. This position will serve as the new position of the object X_i.

Algorithm 21: mapFromComputedCoordinatesToRealPosition() algorithm

Input: c_o,X_i,$N(c_o)$
Output: new X_i

$positions \leftarrow \varnothing$
foreach $v_j \in N(c_o)$ **do**
  $positions \leftarrow positions \cup$ position(V_j)
$V_{nearest} \leftarrow$ getNearestPosition($positions$)
return $V_{nearest}$

Algorithm 22 updates the archive with the new non-dominated solutions NDS. It takes as input the current archive $archive$ and the set of new non-dominated solutions NDS. The algorithm checks if each solution Sol_i dans NDS can be added to the archive by respecting the domination rules. If the archive is empty, the solution Sol_i is added directly to the archive. Otherwise, for each solution S already present in the archive, the algorithm checks if Sol_i dominates it. If Sol_i dominates S, then S is excluded from the archive. If the archive is full after adding Sol_i, the algorithm reorganizes the grid segmentation and excludes a solution from the most-crowded segment to add the new solution Sol_i to the updated archive.
Section 5.5 of the following chapter describes the conducted experiments and the results of evaluating the proposed geographic routing detailed in this section.

4.8 Conclusion

To conclude, this chapter presented the introduced theoretical contributions. We first suggested a mathematical modeling of routing in IoT networks. Then, an improved version of MOGWO is developed . Afterwards, an optimization of the parameters of Deep Learning models using metaheuristics was proposed and its application to develop an approach of indoor localization in

Algorithm 22: updateArchive() algorithm

Input: NDS
Output: updated archive

```
foreach Sol_i ∈ NDS do
    if archive is empty then
        archive ← archive ∪ {Sol_i}
    else
        if ∄S ∈ archive | S ≻ Sol_i then
            foreach S ∈ archive do
                if Sol_i ≻ S then
                    archive ← archive \ S
            if archive is full then
                Reorganize the grid segmentation
                Exclude a solution from the most-crowded segment
            archive ← archive ∪ {Sol_i}
```

IoT networks was described. Finally, we provide a section to illustrate our hybrid and multi-objective geographic routing approach to be deployed in IoT networks having mesh architecture. This approach aligns with Edge computing as the routing decision-making is distributed among the objects of the network. In other words, the routing process does not require a central computer to calculate routes for data transmission within the network. The following chapter will focus on the numerical results obtained by the tests and experiments carried out to validate and evaluate the theoretical contributions presented in this chapter.

Chapter 5

Numerical results, simulations and experiments

Contents

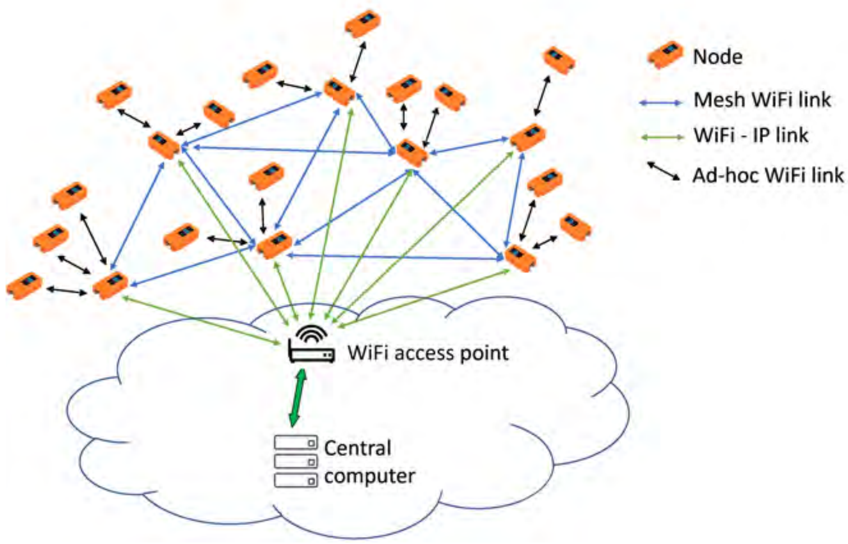
5.1 Introduction

In fact, the previous chapter has initiated our theoretical contributions by presenting the proposed mathematical modelling of routing in IoT networks, accompanied by innovative proposals. In fact, we have introduced a reactive multi-objective routing algorithm based on the MOGWO metaheuristic optimizer. In addition, we have introduced an algorithm, Improved MOGWO, an improved version of the MOGWO, as well as a the method for optimizing the parameters of DL models using metaheuristics. Moreover, we proposed an indoor localization approach combining DL techniques and metaheuristics. Therefore, taking advantage of these approaches, we have developed a hybrid multi-objective geographic routing model designed for the IoT networks with a mesh architecture. This chapter is devoted to the presentation and analysis of the carried out experiments to evaluate our contributions. The first section details the experiments carried out to evaluate our proposal for reactive, multi-objective routing based on MOGWO. We analyze the obtained results, highlighting the performance of our approach. The second section focuses on experiments aimed at testing the resolution of the DTLZ reference problems with our IMOGWO proposal. We theoretically evaluate its performance by comparing the obtained results with those of other widely used optimization algorithms. The third section describes the experimental setup of the tests carried out to evaluate the effectiveness of our contributions relating to the optimization of the weights of DL models by metaheuristic algorithms, as well as the development of a DL model for indoor localization based on UWB ToF measurements. In addition, it presents the results obtained during the evaluation of these approaches and compares them with those of other indoor localization solutions. Finally, the last section presents and analyses the results of experiments on our geographical and hybrid routing proposal. We evaluate the performance of this solution, comparing it with other existing routing solutions, and test the efficiency of the IMOGWO algorithm by applying it to a real routing problem, outside the DTLZ reference problems.

5.2 Experimental results of the MOGWO-based routing

In fact, this section is devoted to the presentation of the experiments carried out to evaluate our contribution to reactive and multi-objective routing based on the MOGWO metaheuristic optimization algorithm, as detailed in section 4.3, and to analyzing the obtained results.

Figure 5.1: Proposed network architecture



5.2.1 Details of the experiments

Table 5.21 shows the parameters of the experiments carried out to evaluate the proposed routing algorithm. These experiments were carried out on an IoT network containing recent Wi-Fi/BLE 2.4 GHz connected objects called M5StickC [143]. In the experimental configuration, 22 M5StickC objects were deployed on a 300*300 m^2 surface. These objects are IoT devices based on an Arduino-compatible ESP32 microprocessor. The M5StickC node is equipped with an infrared transmitter, a microphone, an LCD screen and a 4 MB internal memory. To operate these M5StickC nodes, we used the Arduino Painless Mesh library [144], which, unlike the ESP-NOW library [145] used in [146], provides access to a series of wireless frame transmission and reception functions, in a topology that can be based on a tree hierarchy. Also, ESP-NOW requires that all nodes must be within radio range of each other, as there is no multi-hop routing provided [146]. Experimental parameters include the average number of executions, set at 25, and the communication technology used, which is Wi-Fi. The transmission/detection range is approximately 27 meters on average. The transmission power used for the experiments is 100 mW. A frame exchange scenario between random nodes in the network is used to test the quality of links and routing.
Figure 5.1 represents the proposed network architecture.

Table 5.1: The parameters of the experiments

Average number of executions	25
Area	300 * 300 m^2
Number of M5StickC devices	22 fixed objects and 1 mobile object
Communication technology	Wi-Fi
Transmission/Detection range	27 m (average value)
RSSI	Variable (initialized to 120 milliwatt)
Frame Error Rate (FER)	Variable (initialized to 0.01)
Transmission power	100 mW

5.2.2 The comparative routing algorithms

In this vein, we evaluate and compare our proposed routing algorithm applied with MOGWO against a recent multi-objective algorithm, NSGA-III [59]. Table 5.2 shows the parameters used for the two multi-objective metaheuristics applied, MOGWO and NSGA-III.

Table 5.2: The parameters of MOGWO and NSGA-III

<table>
<tr><th colspan="3">Parameter</th><th>Value</th></tr>
<tr><td colspan="3">Number of executions</td><td>25, using distinct initial populations</td></tr>
<tr><td colspan="3">Number of generations</td><td>10 to 650</td></tr>
<tr><td colspan="3">Population size</td><td>300</td></tr>
<tr><td colspan="3">Number of objectives</td><td>3 to 5</td></tr>
<tr><td rowspan="4">MOGWO</td><td colspan="2">Archive size</td><td>50</td></tr>
<tr><td colspan="2">alpha</td><td>0.1</td></tr>
<tr><td colspan="2">beta</td><td>4</td></tr>
<tr><td colspan="2">gamma</td><td>2</td></tr>
<tr><td rowspan="6">NSGA-III</td><td rowspan="3">Mutation</td><td>Operator</td><td>Swap Mutation</td></tr>
<tr><td>Distribution index</td><td>45</td></tr>
<tr><td>Probability</td><td>0.0025</td></tr>
<tr><td rowspan="3">Crossover</td><td>Operator</td><td>Sequential Constructive (SCX)</td></tr>
<tr><td>Distribution index</td><td>25</td></tr>
<tr><td>Probability</td><td>0.9</td></tr>
</table>

5.2.3 The evaluation criteria

Therefore, in order to evaluate the behavior of the multi-objective optimizers applied to develop our proposed routing approach, the following evaluation criteria are used:

- The HV (see section 3.2.5) is used to evaluate the convergence and distribution of the solutions generated by the two compared optimization algorithms;
- The average lifetime of the network expresses the capacity of the routing solutions to offer efficient optimization of energy consumption and to maintain a load balance between the objects in the network;

- The average number of neighbors of the objects in the network to evaluate the capacity of the two compared routing solutions to maintain a good distribution of the objects in the studied IoT environment;
- The transmission delay between the initial source and the final destination;
- The average RSSI value, which represents the received signal and its quality.

5.2.4 Results and discussion

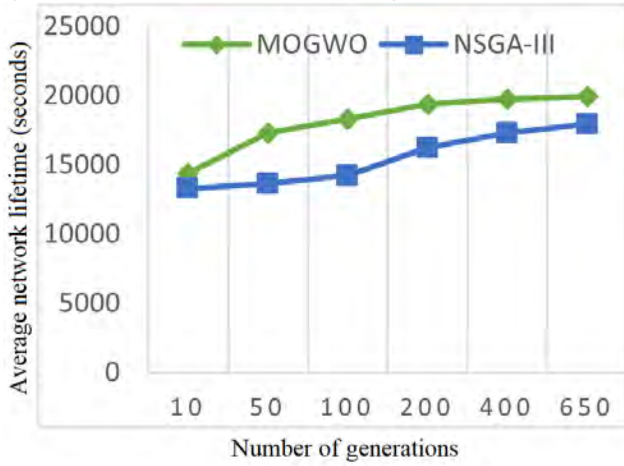
Table 5.3 presents the results in terms of the Hypervolume indicator obtained by the two routing algorithms compared. Three test scenarios were run, varying the number of generations and the number of considered objectives. In the first scenario, with 350 generations and a single objective, the MOGWO outperformed NSGA-III in terms of best and worst performance in Hypervolume, with 0.921594 and 0.901168 respectively. However, NSGA-III outperformed MOGWO in terms of average value. Then, in the second scenario, with 450 generations and two objectives, MOGWO also outperformed NSGA-III for the best and worst performance cases, with 0.982056 and 0.979087 respectively. However, for the mean value, NSGA-III showed a slight superiority. In the third scenario with 650 generations and three objectives, MOGWO maintained its superiority by outperforming with a Hypervolume of 0.973267, while NSGA-III achieved 0.921185. In addition, MOGWO also outperformed NSGA-III in terms of mean value, although in worst-case, performance are almost identical. In conclusion, the results indicate that MOGWO generally performs better in terms of Hypervolume than NSGA-III in different test situations.

Table 5.3: Hypervolume indicator values for MOGWO and NSGA-III applied to multi-objective routing problem

Number of generations	Number of objectives	Value	MOGWO	NSGA-III
350	1	Best	**0.921594**	0.921185
		Average	0.914165	**0.916874**
		Worst	**0.901168**	0.881268
450	2	Best	**0.982056**	0.981134
		Average	0.980005	**0.981023**
		Worst	**0.979087**	0.978096
650	3	Best	**0.973267**	0.921185
		Average	**0.972601**	0.916874
		Worst	**0.972503**	0.972232

In fact, Figure 5.2 compares the two routing algorithms in terms of the average network lifetime obtained for different numbers of executed generations. Overall, we observe an increase in the average lifetime of the network with an increasing number of generations for both algorithms. However, there are significant differences between the two approaches. As for a routing with the MOGWO,

Figure 5.2: The average network lifetime for routing modeled with MOGWO and NSGA-III



a relatively constant increase in the average lifetime of the network is observed as the number of generations increases, illustrating the capacity of this algorithm to maintain a gradual improvement in the energy performance of the network. On the other hand, for routing with NSGA-III, although the average network lifetime also increases with the number of generations, the values are generally lower than those obtained with the MOGWO. This shows that, despite increasing optimization as the number of generations increases, NSGA-III fails to match the MOGWO in terms of energy efficiency and maintaining a load balance between the network objects. In summary, the results indicate that the MOGWO provided a better opportunity to optimize the energy consumption and maintain the average lifetime of the network compared with NSGA-III.

Indeed, Figure 5.3 compares the two algorithms in terms of the average number of neighbors of objects in the network. The results shown indicate that, on average, the number of object neighbors when routing with MOGWO is higher than NSGA-III for each number of generations considered, indicating that MOGWO-based routing tends to maintain a better distribution of objects in the studied IoT environment, which could contribute to better efficiency and reduced network congestion.

On the other hand, Figure 5.4 presents the evaluation results of two algorithms compared in terms of transmission delay between the initial source and the final destination. In fact, an analysis of the results shows that, on average, the transmission delay is shorter when the routing is carried out with the MOGWO than with NSGA-III for each considered number of generations. This capacity to offer shorter transmission times shows a better efficiency in data transmission when the MOGWO is used as a multi-objective optimizer to develop the proposed routing algorithm, which may contribute to better network reactivity and faster data delivery.

In addition, Figure 5.5 presents the mean RSSI values obtained by the two routing algorithms evaluated for the different considered number of generations. The values have been converted to milliwatts. In fact, the analysis of the presented results shows that, on average, the RSSI value is

Figure 5.3: The average number of neighbors for routing modeled with MOGWO and NSGA-III

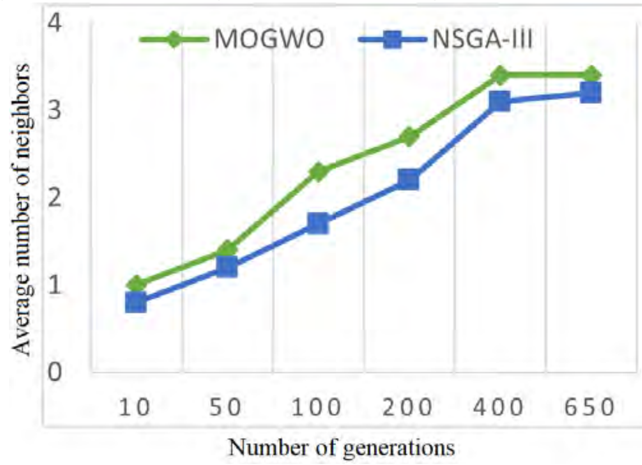


Figure 5.4: End-to-end transmission delay for routing modeled with MOGWO and NSGA-III

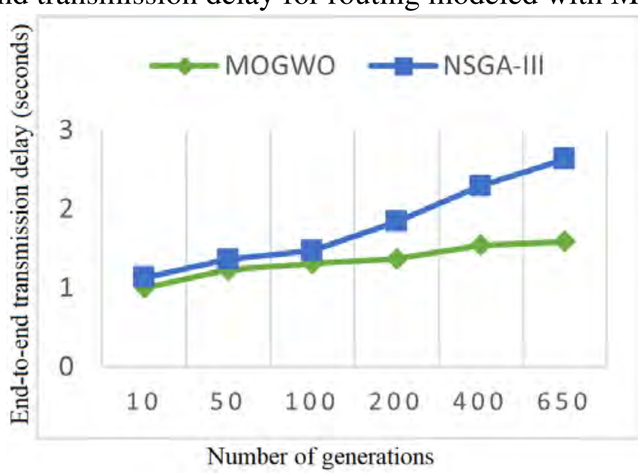
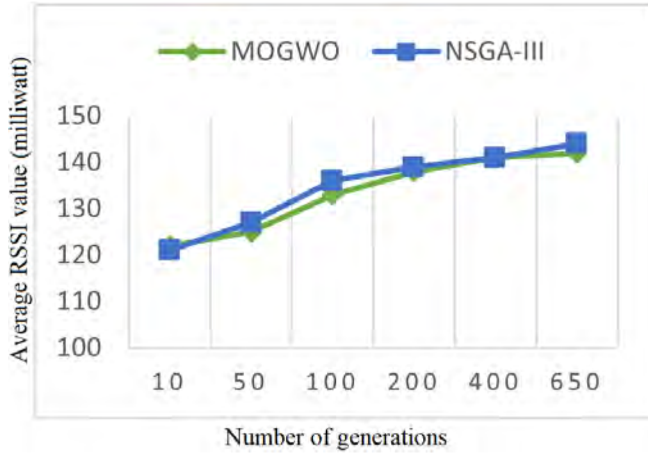


Figure 5.5: The average RSSI value for routing modeled with MOGWO and NSGA-III



slightly high when the routing is performed with the MOGWO compared to NSGA-III for each considered number of generations. This also shows that the MOGWO tends to offer better signal quality in terms of receiving strength than NSGA-III, thus ensuring more reliable and stable communication in the studied IoT network.

5.2.5 Synthesis

Indeed, the comparative analysis between the two multi-object routing algorithms for the hybrid topology IoT networks, which are based on the MOGWO and NSGA-III, revealed that the MOGWO generally outperforms NSGA-III on several criteria. In fact, the MOGWO demonstrates better performance in terms of Hypervolume, average network lifetime, the number of object neighbors, transmission delay and RSSI signal quality. In addition, these results indicate that the routing proposed with the MOGWO offers a more efficient and promising solution for the routing in the IoT networks, increased energy efficiency, improved load balancing, reduced transmission delays, more reliable and stable communications and the maintenance of an efficient distribution of objects in the network.
This contribution and its results are the subject of our publication [138].

5.3 IMOGWO numerical results on DTLZ test problems

In order to theoretically evaluate the performance of the proposed IMOGWO algorithm, experiments were conducted to test the resolution of the DTLZ benchmark problems. The obtained results are compared with those obtained by recent and widely used optimization algorithms. To do this, we used jMetal [147], a framework written in Java dedicated to multi-objective optimization, to calculate the performance indicators. The experiments were run on the computing servers of the Computer Science Research Institute of Toulouse (Institut de Recherche en Informatique de Toulouse (IRIT)) [148], platforms providing access to high-performance computing resources. This section details the carried out experiments and analyses the obtained results. Moreover, statistical tests have been applied to compare the efficiency of the proposed optimizer with that of the other studied optimizers.

5.3.1 Test details

In fact, this section presents the used algorithms and the standard test problems, as well as the performance indicators used to compare and numerically evaluate the studied algorithms.

5.3.1.1 Comparison algorithms and configuration parameters

The proposed algorithm (IMOGWO) has been compared to the MOGWO as well as to other recent and well-known multi-objective optimization algorithms: MOEA/D-IEps [69], MOEA/DD [66], ESPEA [64] and DMOPSO [63]. The initial parameters used by the studied algorithms are presented in Table 5.4. The experiments were carried out using the jMetal framework [147], which uses automatic tuning methods to configure the algorithms and determine the typical parameters. The initial parameters of DMOPSO, MOEA/D-IEps, MOEA/DD and ESPEA are therefore the same as those used in jMetal. Moreover, the MOGWO parameters correspond to those mentioned in the study that proposed it, [2], where the population size and the number of generations were fixed at 100 and 3000, respectively. Therefore, to ensure a fair comparison between the studied optimizers, these same values were used for the other algorithms. Thus, a population of 100 individuals was used for all tests, with randomly generated elements respecting the domains of the variables in each problem. The number of generations for each algorithm was 3000. All algorithms were run 30 times on the test problems.

Table 5.4: Initial parameters used by the studied algorithms

Algorithm	IMOGWO	MOGWO	DMOPSO	MOEA/D-IEps	MOEA/DD	ESPEA
Parameters	Number of bisections = 10	Number of bisections = 10	r1, r2: random real numbers between 0.0 and 0.1	Neighborhood size: 30	Neighborhood size: 20	Crossover probability = 0.9
			Cognitive, social parameters: random real numbers between 1.5 and 2.5	Neighborhood selection probability: 0.9	Neighborhood selection probability: 0.9	Crossover distribution index = 20.0
			Inertia weight: 0.4			
	Population size = 100 Number of generations = 3000 Number of executions = 30					

5.3.1.2 The reference problems

To evaluate the IMOGWO efficiency, as a multi-objective optimizer, it was tested on the DTLZ reference problems [4]. These are among the most difficult and popular multi-objective reference problems. They are scalable because they allow the number of variables and objectives to vary. In addition, they offer different search spaces with various forms and characteristics of the PF_{true}: linear, multimodal, concave, biased, mixed, disconnected and scaled. The resolution of problems with different characteristics (DTLZ1-4,7) is evaluated in this study. They are solved with a number of objectives equal to 2, 3, 4, 6 and 8. Table 5.5 describes the characteristics of the examined problems. The mathematical formulas applied in the DTLZ1-4,7 problems and their parameters are represented in Table 5.6.

Table 5.5: Characteristics of the studied DTLZ1-4,7 problems

Problem	**Characteristics**
DTLZ1	Linear, multimodal
DTLZ2	Concave
DTLZ3	Concave, multimodal
DTLZ4	Concave, biased
DTLZ7	Mixed, disconnected, multimodal, scaled

Table 5.6: Mathematical formulas and parameters of the studied DTLZ1-4,7 problems

Problem	Mathematical Formulas	Parameters	Domains	Dimension
DTLZ1	$f_1 = (1+g)0.5\prod_{i=1}^{M-1} x_i$ $f_{m=2:M-1} = (1+g)0.5(\prod_{i=1}^{M-m} x_i)(1-x_{M-m+1})$ $f_M = (1+g)0.5(1-x_1)$ $g = 100[k+\sum_{i=1}^{k}((x_i-0.5)^2 - \cos(20\pi(x_i-0.5)))]$	$k=5$	$0 \leq x_i \leq 1$ $i = 1, 2, ..., n$ $M \in 2, 3, 4, 6, 8$	$n = k+M-1$
DTLZ2	$f_1 = (1+g)\prod_{i=1}^{M-1}\cos(x_i\pi/2)$ $f_{m=2:M-1} = (1+g)(\prod_{i=1}^{M-m}\cos(x_i\pi/2))\sin(y_{M-m+1}\pi/2)$ $f_M = (1+g)\sin(x_1\pi/2)$ $g = \sum_{i=1}^{k}(x_i-0.5)^2$	$k=10$		
DTLZ3	Same $f_{1:M}$ as DTLZ2 $g = 100[k+\sum_{i=1}^{k}((x_i-0.5)^2 - \cos(20\pi(x_i-0.5)))]$	$k=10$		
DTLZ4	Same equations as DTLZ2, except all x_i are replaced by x_i^{α}	$k=10$ $\alpha=100$		
DTLZ7	$f_{m=1:M-1} = x_m$ $f_M = (1+g)(M-\sum_{i=1}^{M-1}[f_i/(1+g)(1+\sin(3\pi f_i))])$ $g = 1+9\sum_{i=1}^{k} x_i/k$	$k=20$		

5.3.1.3 Performance indicators

IGD and NHV (section 3.2.5 of chapter 3) have been selected for their wide use as evaluation criteria, providing a complete view of the convergence and distribution performance of PF [76]. These criteria are therefore appropriate for evaluating an optimisation algorithm.

5.3.2 Results and discussion

In this section, we present the results of the numerical evaluation of the algorithms studied and applied to the resolution of DTLZ1-4,7 problems. The IGD and NHV values obtained are plotted and compared.

Tables 5.7, 5.8, 5.9, 5.10 and 5.11 show the mean, the median, the best (minimum) and the worst (maximum) values provided by the IGD and NHV indicators after solving the DTLZ1-4,7 problems for each number of objectives. They also include the standard deviation of the different IGD and NHV values obtained during the 30 carried out tests.

These values represent the experimental results obtained by applying IMOGWO and the other comparative algorithms. We use the average values of the two indicators to compare the performance of the optimizers in this section. The best average values of IGD and NHV between those given by IMOGWO and MOGWO are underlined. However, the best average values among those obtained by all the optimizers for each problem and each number of objectives are highlighted in bold.

On the other hand, the experimental results confirm that the IMOGWO outperformed the MOGWO in all the considered test cases, (DTLZ1-4,7 problems) and for all the configurations of the studied objectives (2, 3, 4, 6, 8). Moreover, the values of the IGD and the NHV obtained with the IMOGWO are lower than those obtained with the MOGWO, thus indicating a better convergence and distribution of the Pareto fronts with the IMOGWO.

In fact, this improvement is clearly visible in the curves showing the variation in IGD and NHV for each problem. These curves, illustrated in Figures 5.6 and 5.7, show the average values of IGD and NHV as a function of the number of objectives. In each DTLZ1-4,7 problem, Figure 5.6 comprises two graphs (a) and (b). Graph (a) shows the variation in IGD values for all the algorithms studied. ESPEA's results are higher than those of the other optimizers, making it difficult to visualize the graph, while graph (b) excludes the values provided by ESPEA. Furthermore, Tables 5.7, 5.8, 5.9, 5.10 and 5.11 indicate that IMOGWO generally has the best average values for the IGD and the NHV indicators.

For the DTLZ1 problem, the IMOGWO shows better IGD results for all the configurations of the number of the examined objectives (Table 5.7 and Figure 5.6). Moreover, it has a lower NHV than all the other optimizers for a number of objectives equal to 2, 3, 4 and 6 (Table 5.7 and Figure 5.7). On the other hand, for a number of objectives equal to 8, MOEA/DD obtains the best average NHV value. IMOGWO's ability to provide efficient convergence and distribution is highlighted in Figure 5.8. This representation shows the $PF\text{s}_{true}$ as well as the solutions obtained (the PFs) using IMOGWO on DTLZ1 for numbers of objectives ranging from 2 to 8. It also shows that the PFs obtained remain close to the $PF\text{s}_{true}$ of DTLZ1 despite a deterioration in performance due to the increase in the number of objectives.

Table 5.7: Values of IGD and NHV indicators obtained on the DTLZ1 problem

Algorithm	Value	2 objectives		3 objectives		4 objectives		6 objectives		8 objectives	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Average	**0.000069**	**0.003266**	**0.000378**	**0.036423**	**0.010838**	**0.047032**	**0.019995**	**0.094837**	**0.025651**	0.469739
	Median	0.000069	0.003294	0.000372	0.034945	0.010350	0.038246	0.020218	0.108923	0.025876	0.489081
	Standard Deviation	0.000012	0.000703	0.000040	0.005967	0.001454	0.026641	0.000950	0.062687	0.000410	0.040161
	Worst	0.000096	0.004645	0.000476	0.054440	0.013620	0.098085	0.021141	0.177930	0.025886	0.494559
	Best	0.000050	0.002159	0.000318	0.028707	0.008858	0.013479	0.018106	0.000000	0.024097	0.303328
MOGWO	Average	0.000512	0.029656	0.000860	0.142148	0.018315	0.196963	0.029016	0.425625	0.033925	0.742158
	Median	0.000495	0.029069	0.000847	0.141181	0.018062	0.196306	0.028351	0.419314	0.033131	0.725132
	Standard Deviation	0.000084	0.003412	0.000069	0.012285	0.001277	0.023890	0.003023	0.053751	0.004012	0.070691
	Worst	0.000754	0.037453	0.001021	0.167160	0.022229	0.251157	0.036818	0.578505	0.042183	0.922828
	Best	0.000391	0.024446	0.000748	0.120270	0.015540	0.155524	0.024976	0.329965	0.027762	0.623155
DMOPSO	Average	0.000131	0.009121	0.000603	0.089080	0.016464	0.150495	0.024453	0.196212	0.030945	0.367096
	Median	0.000131	0.009119	0.000603	0.088957	0.016469	0.150716	0.024437	0.199202	0.030398	0.375070
	Standard Deviation	0.000000	0.000009	0.000005	0.002259	0.000262	0.003606	0.000764	0.022247	0.001958	0.070851
	Worst	0.000131	0.009153	0.000614	0.094548	0.016941	0.158234	0.026130	0.235594	0.035192	0.551234
	Best	0.000130	0.009110	0.000594	0.083779	0.015842	0.143151	0.023015	0.148535	0.027090	0.234799
MOEA/D-IEps	Average	0.000138	0.009511	0.000610	0.097178	0.011308	0.080268	0.021080	0.113920	0.030168	0.250805
	Median	0.000138	0.009518	0.000607	0.098252	0.011447	0.079977	0.021055	0.112634	0.030284	0.250758
	Standard Deviation	0.000001	0.000037	0.000030	0.006027	0.000337	0.009793	0.001049	0.039616	0.002007	0.039388
	Worst	0.000138	0.009518	0.000684	0.108379	0.011860	0.102330	0.023428	0.186236	0.033228	0.356863
	Best	0.000134	0.009313	0.000560	0.085434	0.010573	0.057387	0.019006	0.023114	0.024108	0.175882
MOEA/DD	Average	0.000130	0.009111	0.000520	0.058635	0.019925	0.128729	0.022230	0.110635	0.027129	**0.125032**
	Median	0.000130	0.009109	0.000520	0.058634	0.019925	0.128729	0.022272	0.110196	0.027129	0.125027
	Standard Deviation	0.000000	0.000007	0.000000	0.000012	0.000000	0.000002	0.000252	0.002476	0.000002	0.000058
	Worst	0.000131	0.009141	0.000520	0.058656	0.019925	0.128731	0.022415	0.115673	0.027131	0.125149
	Best	0.000130	0.009109	0.000520	0.058612	0.019924	0.128726	0.020974	0.101622	0.027125	0.124900
ESPEA	Average	0.000132	0.009208	0.000506	0.052285	0.019748	0.202768	1.649086	1.000000	59.174857	1.000000
	Median	0.000132	0.009207	0.000503	0.052128	0.019347	0.191180	0.234233	1.000000	58.943879	1.000000
	Standard Deviation	0.000000	0.000008	0.000010	0.000923	0.002255	0.051301	3.155118	0.000000	1.589821	0.000000
	Worst	0.000132	0.009234	0.000535	0.054501	0.026387	0.371539	12.760023	1.000000	62.719514	1.000000
	Best	0.000132	0.009197	0.000494	0.050984	0.016279	0.128001	0.065438	1.000000	55.262377	1.000000

For the DTLZ2 problem, the IGD and NHV results obtained (see Table 5.8, Figure 5.6 and Figure 5.7) clearly demonstrate the superiority of the IMOGWO over the MOGWO for all the numbers of the considered objectives: 2, 3, 4, 6 and 8. Initially, in terms of IGD, the IMOGWO outperforms all the other comparative algorithms for 3 and 4 objectives. However, the best average IGD values were provided by ESPEA for 2 objectives, and MOEA/D-IEps for 6 and 8 objectives. Despite this, the IMOGWO presented the lowest best IGD value for 2 objectives. On the other hand, MOGWO, DMOPSO and MOEA/DD gave the worst results for all numbers of objectives. Then, for the NHV, the IMOGWO turns out to be the best for 4 objectives. However, for 2 and 3 objectives, the ESPEA gave the lowest average NHV values, while the IMOGWO gave the lowest best NHV value for 2 objectives. For 6 and 8 objectives, the NHV results show that the MOEA/DD is the most efficient.

Table 5.8: Values of IGD and NHV indicators obtained on the DTLZ2 problem

Algorithm	Value	2 objectives		3 objectives		4 objectives		6 objectives		8 objectives	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Average	0.000196	0.021344	**0.000604**	0.159711	**0.010474**	**0.166278**	0.021944	0.315858	0.028232	0.360284
	Median	0.000197	0.022155	0.000605	0.161147	0.010319	0.164841	0.022028	0.317608	0.028215	0.358880
	Standard Deviation	0.000031	0.002890	0.000029	0.007295	0.000574	0.016177	0.000513	0.010170	0.000273	0.006830
	Worst	0.000262	0.025427	0.000648	0.175340	0.012347	0.206035	0.022724	0.332887	0.028613	0.371265
	Best	0.000149	0.016349	0.000538	0.144363	0.009559	0.132658	0.020194	0.275735	0.027443	0.344187
MOGWO	Average	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023146	0.343687	0.028614	0.378260
	Median	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023144	0.343757	0.028613	0.378066
	Standard Deviation	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000006	0.000516	0.000002	0.000428
	Worst	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023165	0.344740	0.028623	0.380165
	Best	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023138	0.342908	0.028613	0.378066
DMOPSO	Average	0.000173	0.018370	0.000816	0.208090	0.016303	0.275250	0.026875	0.281056	0.030979	0.357137
	Median	0.000173	0.018364	0.000815	0.207933	0.016304	0.275839	0.026824	0.279158	0.031002	0.346567
	Standard Deviation	0.000000	0.000032	0.000005	0.003880	0.000095	0.001965	0.000372	0.020820	0.001784	0.023997
	Worst	0.000173	0.018435	0.000825	0.215057	0.016482	0.278648	0.027579	0.334046	0.033999	0.398225
	Best	0.000173	0.018318	0.000805	0.200396	0.016081	0.270695	0.026235	0.245394	0.026439	0.325745
MOEA/D-IEps	Average	0.000162	0.018995	0.000790	0.218628	0.011813	0.184438	**0.021052**	0.166039	**0.023672**	0.225824
	Median	0.000162	0.018904	0.000791	0.216210	0.011773	0.187251	0.021078	0.165951	0.023498	0.228383
	Standard Deviation	0.000003	0.000437	0.000033	0.016475	0.000 498	0.020268	0.000536	0.020099	0.000731	0.031303
	Worst	0.000167	0.019969	0.000890	0.266657	0.013204	0.216263	0.022190	0.204615	0.026211	0.278026
	Best	0.000157	0.018338	0.000734	0.183085	0.010797	0.138909	0.019962	0.128972	0.022754	0.159115
MOEA/DD	Average	0.000173	0.018250	0.000667	0.141488	0.019390	0.217154	0.021999	**0.081595**	0.026414	**0.000000**
	Median	0.000173	0.018250	0.000667	0.141493	0.019390	0.217153	0.022012	0.081681	0.026414	0.000000
	Standard Deviation	0.000000	0.000000	0.000000	0.000026	0.000000	0.000002	0.000078	0.000424	0.000000	0.000000
	Worst	0.000173	0.018250	0.000667	0.141513	0.019391	0.217159	0.022102	0.082045	0.026414	0.000000
	Best	0.000173	0.018250	0.000667	0.141352	0.019389	0.217149	0.021591	0.079428	0.026413	0.000000
ESPEA	Average	**0.000157**	**0.017179**	0.000658	**0.129807**	0.015704	0.283479	0.162279	1.000000	0.140856	1.000000
	Median	0.000156	0.017125	0.000657	0.130333	0.015743	0.272516	0.162262	1.000000	0.140841	1.000000
	Standard Deviation	0.000002	0.000267	0.000011	0.004172	0.000936	0.059490	0.000328	0.000000	0.000190	0.000000
	Worst	0.000163	0.017710	0.000681	0.140629	0.017810	0.470555	0.162788	1.000000	0.141209	1.000000
	Best	0.000154	0.016779	0.000638	0.121541	0.014164	0.194287	0.161610	1.000000	0.140434	1.000000

On the other hand, the highest (worst) NHV values were obtained by applying the MOGWO, the DMOPSO and the MOEA/D-IEps for the whole number of objectives. Although IMOGWO is not the best in all the DTLZ2 test cases, the obtained solutions are close to the actual values of the Pareto fronts and are well distributed, as shown in Figure 5.9.

The results obtained with the DTLZ3 (Table 5.9, Figure 5.6 and Figure 5.7) show an improvement in the IGD and NHV using the IMOGWO over the MOGWO in all configuration tests. These results also provide an overview of the performance of all the evaluated optimizers. The IMOGWO particularly stands out for a number of objectives equal to 2, 3 and 8, with the lowest IGD values of all the tested algorithms. For 4 and 6 objectives, the best performing algorithms are the MOEA/D-IEps and the MOEA/DD respectively. In terms of NHV, the IMOGWO offers the best

Table 5.9: Values of IGD and NHV indicators obtained on the DTLZ3 problem

Algorithm	Value	2 objectives		3 objectives		4 objectives		6 objectives		8 objectives	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Average	**0.000066**	**0.004764**	**0.001029**	0.114930	0.012107	0.190469	0.024302	0.327768	**0.025070**	0.312575
	Median	0.000059	0.003877	0.001005	0.099720	0.012250	0.198362	0.024342	0.330156	0.025045	0.308083
	Standard Deviation	0.000020	0.001936	0.000178	0.045068	0.000384	0.025000	0.000150	0.005793	0.000043	0.015273
	Worst	0.000124	0.009849	0.001315	0.188568	0.012394	0.210208	0.024342	0.330156	0.025145	0.368083
	Best	0.000046	0.002798	0.000770	0.062552	0.010975	0.103922	0.023616	0.305431	0.025014	0.303894
MOGWO	Average	0.000447	0.047157	0.001917	0.285572	0.016845	0.350873	0.031653	0.533869	0.034943	0.597588
	Median	0.000454	0.049265	0.001871	0.281219	0.016830	0.358689	0.031934	0.535161	0.034874	0.589864
	Standard Deviation	0.000100	0.010610	0.000205	0.021580	0.001316	0.034478	0.001986	0.027713	0.001919	0.032263
	Worst	0.000666	0.067131	0.002288	0.344266	0.019619	0.393871	0.035103	0.597043	0.038714	0.679805
	Best	0.000342	0.035563	0.001645	0.248529	0.012394	0.210208	0.027506	0.476556	0.031886	0.530805
DMOPSO	Average	0.000157	0.018630	0.001322	0.194173	0.016376	0.270189	0.028300	0.311864	0.027780	0.268741
	Median	0.000157	0.018629	0.001321	0.194394	0.016385	0.270969	0.027765	0.308394	0.027493	0.262826
	Standard Deviation	0.000000	0.000006	0.000011	0.005923	0.000254	0.003497	0.001161	0.031965	0.001055	0.028103
	Worst	0.000157	0.018649	0.001345	0.205136	0.016923	0.276436	0.032129	0.378852	0.030518	0.333739
	Best	0.000156	0.018622	0.001293	0.174917	0.015909	0.262557	0.027107	0.260172	0.025864	0.209681
MOEA/D-IEps	Average	0.000157	0.016965	0.001269	0.201929	**0.011933**	0.180709	0.023010	0.252782	0.025454	0.370433
	Median	0.000157	0.016838	0.001267	0.203506	0.011779	0.177088	0.022889	0.234884	0.025403	0.357255
	Standard Deviation	0.000001	0.000472	0.000059	0.014932	0.000566	0.022061	0.000971	0.061165	0.001143	0.094737
	Worst	0.000160	0.017811	0.001418	0.225260	0.013531	0.226847	0.025096	0.381325	0.028552	0.602770
	Best	0.000155	0.016303	0.001166	0.175337	0.010919	0.141086	0.021565	0.159411	0.023435	0.197009
MOEA/DD	Average	0.000156	0.018605	0.001065	0.134696	0.019390	0.217152	**0.021887**	**0.084542**	0.026414	**0.000000**
	Median	0.000156	0.018604	0.001065	0.134697	0.019390	0.217152	0.021872	0.084002	0.026414	0.000000
	Standard Deviation	0.000000	0.000002	0.000000	0.000009	0.000001	0.000005	0.000051	0.001294	0.000000	0.000000
	Worst	0.000156	0.018617	0.001065	0.134710	0.019392	0.217160	0.022044	0.087948	0.026414	0.000000
	Best	0.000156	0.018604	0.001065	0.134669	0.019386	0.217132	0.021780	0.083700	0.026413	0.000000
ESPEA	Average	0.000148	0.016905	0.001079	**0.103940**	0.015307	**0.168996**	1.079875	1.000000	108.4481	1.000000
	Median	0.000148	0.016888	0.001051	0.103825	0.015344	0.150555	0.309020	1.000000	108.6618	1.000000
	Standard Deviation	0.000001	0.000093	0.000070	0.001568	0.001460	0.057338	1.650527	0.000000	3.111119	0.000000
	Worst	0.000151	0.017111	0.001259	0.106741	0.018228	0.346247	8.286368	1.000000	112.9668	1.000000
	Best	0.000145	0.016702	0.001003	0.101416	0.013054	0.098215	0.085237	1.000000	102.2059	1.000000

performance for 2 objectives, while the ESPEA dominates for 3 and 4 objectives. For 6 and 8 objectives, the MOEA/DD is the most efficient. The distribution of Pareto fronts obtained by the IMOGWO after solving the DTLZ3, is presented in Figure 5.10, highlighting the IMOGWO's ability to efficiently converge to PF_{true} and to maintain an appropriate distribution of solutions, in particular for 2, 3, 4 and 6 objectives.

The fourth problem, DTLZ4, highlights the performance of the IMOGWO. The PF_{true} for numbers of objectives of 2, 3, 6 and 8 are illustrated in Figure 5.11. Their comparison with the results obtained by the IMOGWO shows satisfactory convergence and distribution in most cases. The IGD and NHV values and their evolution are presented in Table 5.10, Figure 5.6 and Figure 5.7. The IMOGWO outperforms the MOGWO in terms of IGD and NHV in all the tested scenarios. In

Table 5.10: Values of IGD and NHV indicators obtained on the DTLZ4 problem

Algorithm	Value	2 objectives		3 objectives		4 objectives		6 objectives		8 objectives	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Average	0.000506	0.054129	0.001684	0.360809	0.019157	0.319157	0.037798	0.633899	**0.032461**	0.635780
	Median	0.000451	0.051394	0.001711	0.371769	0.019393	0.284616	0.038530	0.656723	0.032051	0.661102
	Standard Deviation	0.000178	0.014713	0.000322	0.099573	0.001315	0.087121	0.002542	0.124194	0.001670	0.116725
	Worst	0.001197	0.095778	0.002323	0.492429	0.021256	0.533864	0.040638	0.813541	0.036879	0.772086
	Best	0.000356	0.034310	0.000940	0.142299	0.016527	0.213991	0.029180	0.390138	0.030236	0.282356
MOGWO	Average	0.004026	0.341989	0.002735	0.504163	0.042100	0.643649	0.053683	0.814735	0.046146	0.772086
	Median	0.004026	0.341989	0.002735	0.504163	0.042099	0.643604	0.053671	0.813936	0.046149	0.772086
	Standard Deviation	0.000000	0.000000	0.000000	0.000000	0.000001	0.000204	0.000180	0.001360	0.000006	0.000000
	Worst	0.004026	0.341989	0.002735	0.504163	0.042102	0.644594	0.054053	0.817091	0.046154	0.772086
	Best	0.004026	0.341989	0.002735	0.504163	0.042099	0.643521	0.053482	0.813626	0.046137	0.772086
DMOPSO	Average	0.000161	0.019825	0.001002	0.114134	0.024345	0.311998	0.040914	0.453365	0.040932	0.493325
	Median	0.000160	0.019792	0.000994	0.112846	0.024118	0.305437	0.040922	0.450765	0.041036	0.495337
	Standard Deviation	0.000002	0.000236	0.000066	0.009308	0.001305	0.021361	0.001952	0.037820	0.001369	0.043626
	Worst	0.000166	0.020302	0.001157	0.139359	0.027239	0.366492	0.045388	0.537338	0.044053	0.587461
	Best	0.000159	0.019321	0.000858	0.098928	0.022119	0.286135	0.037274	0.374113	0.038430	0.400545
MOEA/D-IEps	Average	0.000163	0.019675	0.001457	0.138597	0.021770	**0.155166**	0.033358	0.131438	0.037714	0.136173
	Median	0.000163	0.019610	0.001481	0.139313	0.021789	0.157871	0.034416	0.118508	0.037615	0.132807
	Standard Deviation	0.000002	0.000392	0.000117	0.018923	0.001745	0.014225	0. 003220	0.038455	0.001012	0.021995
	Worst	0.000167	0.020643	0.001690	0.195115	0.024814	0.177411	0.038190	0.196062	0.039941	0.180964
	Best	0.000160	0.018942	0.001280	0.103931	0.018104	0.126861	0.026377	0.074517	0.035624	0.102402
MOEA/DD	Average	0.000156	0.018604	0.000988	0.060576	**0.019090**	0.217353	**0.033271**	**0.123987**	0.035214	**0.000000**
	Median	0.000156	0.018604	0.000988	0.060579	0.019090	0.217353	0.035656	0.129790	0.036414	0.000000
	Standard Deviation	0.000000	0.000000	0.000000	0.000010	0.000000	0.000002	0.004304	0.014266	0.002400	0.000000
	Worst	0.000156	0.018604	0.000988	0.060597	0.019092	0.217357	0.039953	0.129792	0.037414	0.000000
	Best	0.000156	0.018604	0.000988	0.060555	0.019089	0.217349	0.025946	0.076907	0.031414	0.000000
ESPEA	Average	**0.000150**	**0.017512**	**0.000609**	**0.051805**	0.019296	0.226526	0.162201	1.000000	0.141011	1.000000
	Median	0.000150	0.017502	0.000611	0.052796	0.019342	0.216586	0.162373	1.000000	0.141010	1.000000
	Standard Deviation	0.000001	0.000265	0.000012	0.005050	0.000459	0.033544	0.000538	0.000000	0.000183	0.000000
	Worst	0.000152	0.018299	0.000626	0.064094	0.019999	0.297932	0.162887	1.000000	0.141368	1.000000
	Best	0.000148	0.017008	0.000581	0.042291	0.018197	0.176084	0.160633	1.000000	0.140635	1.000000

particular, the IMOGWO stands out as the most efficient optimizer for the largest number of the considered objectives, 8. As for the DTLZ4 with 4 objectives, the IMOGWO shows the best performance in terms of IGD. However, its performance is weaker for the other numbers of objectives. The ESPEA shows the lowest IGD and the NHV values for 2 and 3 objectives. For 4 objectives, the best IGD and NHV values are provided respectively by the MOEA/DD and MOEA/D-IEps. With 6 objectives, the MOEA/DD offers the best values for the evaluated indicators. Similarly, the MOEA/DD has the lowest NHV value for 8 objectives.

The last problem examined in our experiments is the DTLZ7, which, with its mixed disconnected, multimodal and scaled PF_{true}, represents an additional challenge in terms of complexity. The IGD and NHV results presented in Table 5.11 and illustrated in Figures 5.6 and 5.7 confirm the

Table 5.11: Values of IGD and NHV indicators obtained on the DTLZ7 problem

Algorithm	Value	2 objectives		3 objectives		4 objectives		6 objectives		8 objectives	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Average	**0.000075**	**0.000000**	0.002977	**0.067898**	**0.008803**	**0.040202**	**0.026597**	**0.156265**	**0.027944**	**0.347142**
	Median	0.000063	0.000000	0.003080	0.070068	0.008807	0.041640	0.026541	0.154124	0.027800	0.331857
	Standard Deviation	0.000032	0.000000	0.000256	0.010772	0.000185	0.019083	0.000279	0.054959	0.000640	0.072792
	Worst	0.000153	0.000000	0.003191	0.085439	0.009348	0.076408	0.027211	0.290927	0.030073	0.519336
	Best	0.000039	0.000000	0.001743	0.014402	0.008481	0.004729	0.025977	0.054205	0.027374	0.220474
MOGWO	Average	0.002565	0.027961	0.003732	0.197622	0.011734	0.339720	0.028639	0.528310	0.030080	0.703684
	Median	0.002565	0.027961	0.003732	0.197622	0.011734	0.339720	0.028638	0.528310	0.030067	0.703684
	Standard Deviation	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000007	0.000000	0.000064	0.000000
	Worst	0.002565	0.027961	0.003732	0.197622	0.011734	0.339720	0.028665	0.528310	0.030186	0.703684
	Best	0.002565	0.027961	0.003732	0.197622	0.011734	0.339720	0.028638	0.528310	0.029987	0.703684
DMOPSO	Average	0.000591	0.000656	0.004231	0.203412	0.025798	0.906156	0.048252	0.926579	0.035591	0.948974
	Median	0.000591	0.000656	0.004225	0.203562	0.025799	0.906894	0.048271	0.927802	0.035577	0.949154
	Standard Deviation	0.000000	0.000000	0.000033	0.000844	0.000056	0.005137	0.000231	0.006002	0.000324	0.008123
	Worst	0.000592	0.000656	0.004296	0.205004	0.025912	0.917624	0.048665	0.936626	0.036230	0.964524
	Best	0.000591	0.000655	0.004184	0.201797	0.025665	0.897218	0.047622	0.913974	0.034890	0.933942
MOEA/D-IEps	Average	0.000435	0.000001	0.002953	0.134732	0.012622	0.490871	0.032235	0.698074	0.033578	0.830980
	Median	0.000436	0.000000	0.002945	0.135302	0.012629	0.496097	0.032411	0.706898	0.033617	0.832934
	Standard Deviation	0.000023	0.000394	0.000280	0.011131	0.000589	0.061844	0.001162	0.070552	0.000731	0.035674
	Worst	0.000488	0.000123	0.003590	0.154704	0.013734	0.640475	0.034811	0.836728	0.035176	0.921365
	Best	0.000385	0.000000	0.002454	0.117246	0.010893	0.348397	0.029965	0.524652	0.032224	0.754187
MOEA/DD	Average	0.000587	0.045160	0.003075	0.192114	0.009556	0.364360	0.026937	0.556088	0.033662	0.823738
	Median	0.000580	0.041103	0.003070	0.194204	0.009590	0.365470	0.026950	0.558209	0.033623	0.822294
	Standard Deviation	0.000024	0.025350	0.000110	0.012658	0.000225	0.024079	0.000284	0.006437	0.000251	0.004836
	Worst	0.000637	0.095674	0.003331	0.212505	0.010035	0.405992	0.027447	0.559271	0.034266	0.841971
	Best	0.000556	0.003625	0.002876	0.168567	0.009104	0.285771	0.026350	0.531108	0.033137	0.821053
ESPEA	Average	0.000348	0.000055	**0.001857**	0.081561	0.012169	0.485477	0.042697	1.000000	0.043315	1.000000
	Median	0.000344	0.000000	0.001860	0.081858	0.012139	0.484295	0.042757	1.000000	0.043252	1.000000
	Standard Deviation	0.000013	0.000000	0.000045	0.004148	0.000476	0.048502	0.000784	0.000000	0.000598	0.000000
	Worst	0.000381	0.000000	0.001939	0.091853	0.013186	0.579863	0.044148	1.000000	0.044661	1.000000
	Best	0.000330	0.000000	0.001768	0.074533	0.011271	0.398269	0.040635	1.000000	0.041938	1.000000

superiority of IMOGWO in solving DTLZ7 compared to other optimizers. Firstly, the IMOGWO clearly outperforms the MOGWO in terms of IGD and NHV indicators for all test cases. Secondly, compared to the other methods, the IMOGWO provides the lowest (best) IGD values for 2, 4, 6 and 8 objectives. For 3 objectives, ESPEA obtains the best average IGD. On the other hand, the IMOGWO has the lowest (best) IGD value for 3 objectives. The NHV results also confirm the performance of the IMOGWO, which provides the lowest values for all the numbers of considered objectives. However, the challenges posed by the DTLZ7's PF_{true} are illustrated in Figure 5.12. The results show that despite the complexity of the problem, the IMOGWO manages to maintain high convergence, although its distribution decreases slightly for 2 and 3 objectives.

Figure 5.6: Average IGD values for different numbers of objectives and different reference problems

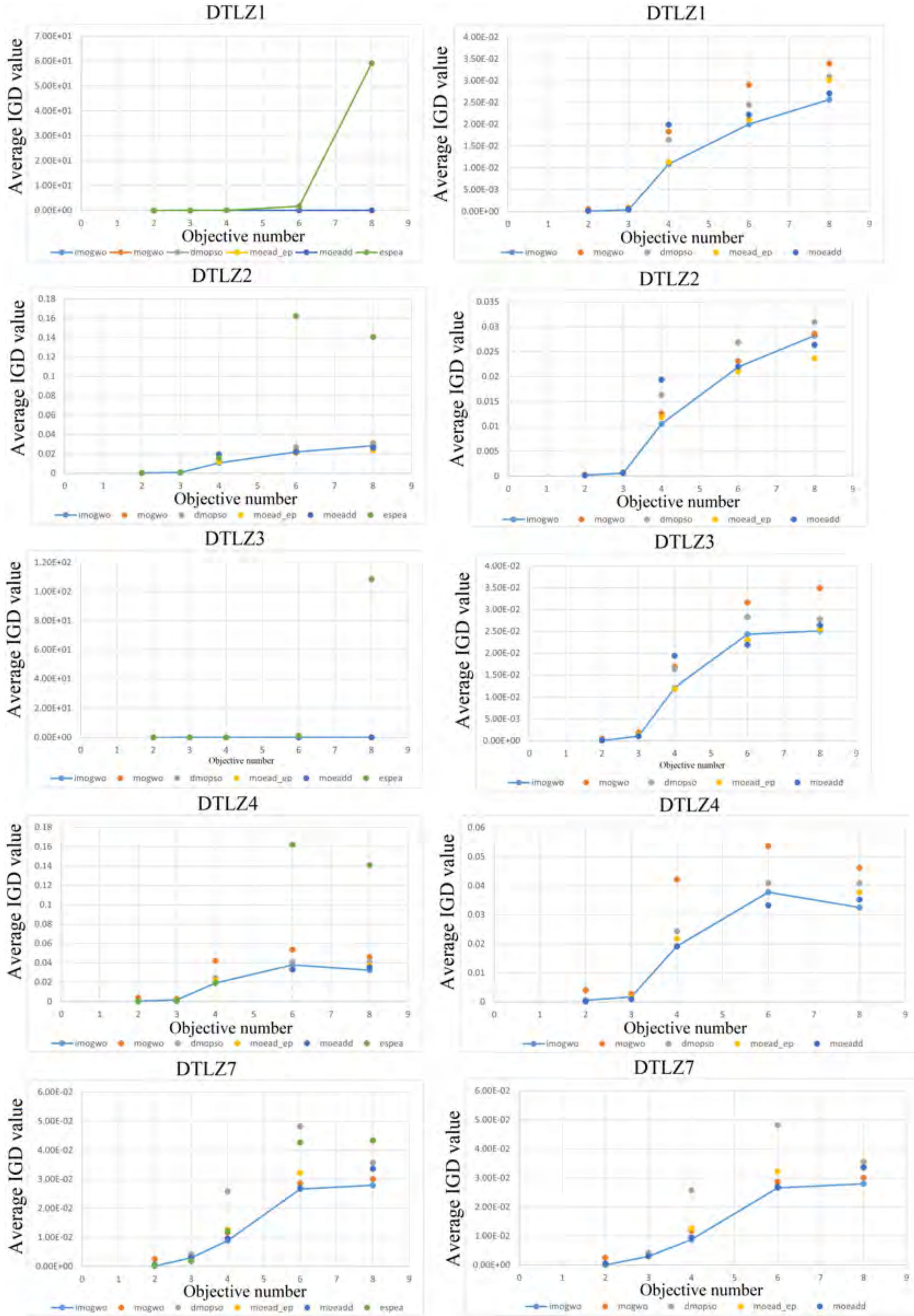
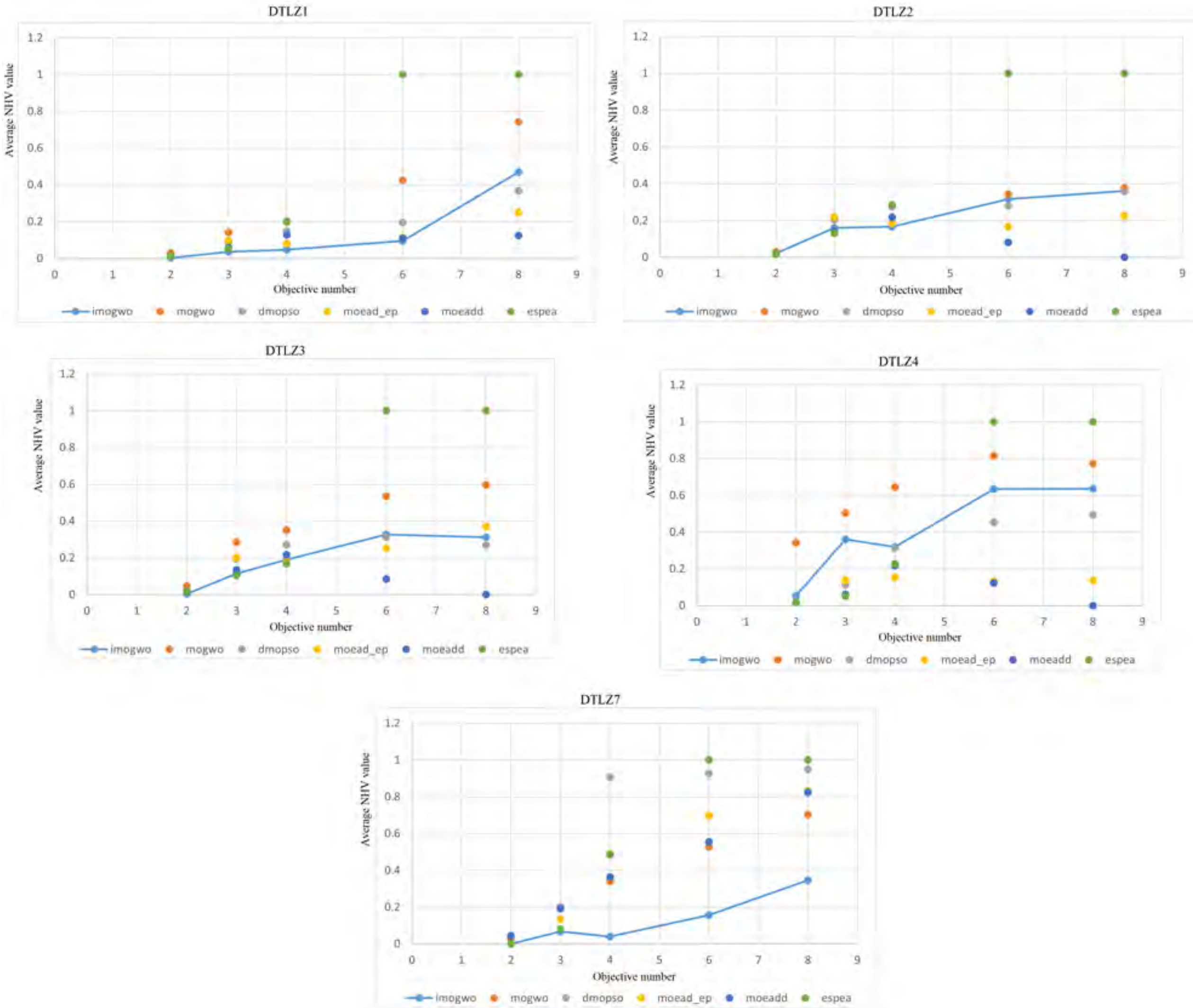
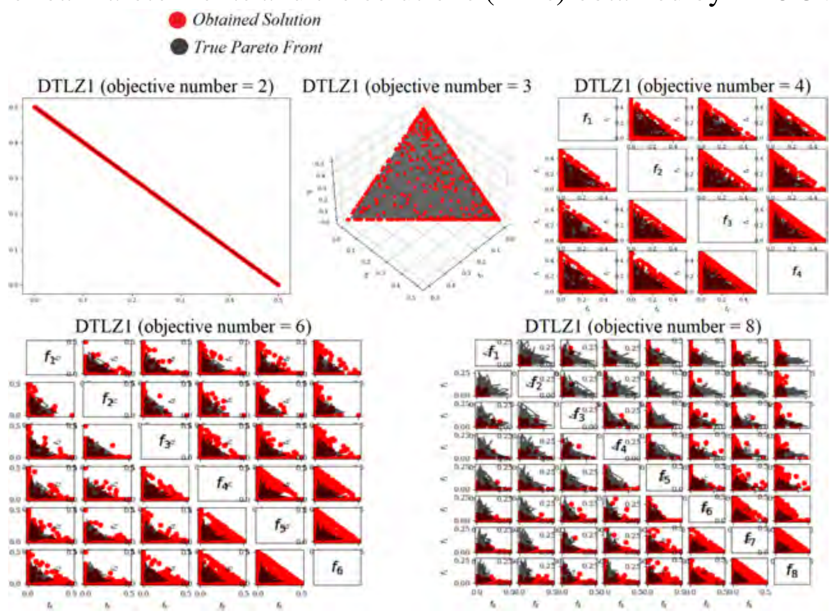


Figure 5.7: Average NHV values for different numbers of objectives and different reference problems



Therefore, to better illustrate the obtained results, we have Figure 5.13 which compares the average values of the IGD and the NHV indicators obtained by the optimization algorithms tested for solving all the DTLZ1-4,7 problems with different numbers of considered objectives. In fact, the IGD indicator provides a complete view of the convergence and distribution performance of a multi-objective optimizer. On the other hand, the IMOGWO stands out with the lowest average IGD value, showing its capacity to deliver competitive and promising results. As for the MOEA/D-IEps, the MOEA/DD and the ESPEA, they have similar values, outperforming the DMOPSO and the MOGWO, which have the worst average IGD performance. On the other hand, the MOEA/DD offers the best overall average performance of the NHV, followed by the IMOGWO in the third place, while the DMOPSO and the MOGWO are ranked fourth and fifth, respectively. Finally, the ESPEA has the highest overall average value for the NHV.

Figure 5.8: The real Pareto fronts and the solutions (PFs) obtained by IMOGWO for DTLZ1



Moreover, a comparison of the optimisers' performance is presented in Figure 5.14, which shows the experimental results obtained using the IGD and NHV indicators for all the DTLZ1-4,7 problems, as well as for the different studied objective configurations. This figure is divided into 3 separate graphs. The first graph (a) in Figure 5.14 shows the overall IGD results obtained by applying all the optimizers in each test scenario. Then graph (b) shows these results while excluding the ESPEA, whose high IGD values make the visualization less clear. Finally, graph (c) highlights the overall NHV results for all the algorithms studied in each test configuration. Then, the used box plots highlight the significant improvement made by the IMOGWO over the MOGWO, both in terms of IGD and NHV, thus enhancing the convergence and distribution of solutions. Compared to other optimizers, the IMOGWO is the best in terms of mean and median IGD values. Similarly, its minimum, median and average NHV values are among the best despite competition from other optimizers.

Consequently, from these graphical representations, it is possible to conclude that, although the IMOGWO algorithm is not always the best in terms of performance on the studied benchmark problems, it is the best in the majority of the carried out tests, particularly for DTLZ1 and DTLZ7. In fact, Table 5.12 summarizes the number of times each algorithm provided the best average IGD and NHV values, given that the total number of carried out test cases (varying the problem and the number of objectives) is 25 (5 different problems * 5 different numbers of objectives). The IMOGWO recorded the best results in terms of IGD and NHV in 15 and 11 cases out of 25, respectively.

Figure 5.9: The real Pareto fronts and the solutions (PFs) obtained by IMOGWO for DTLZ2

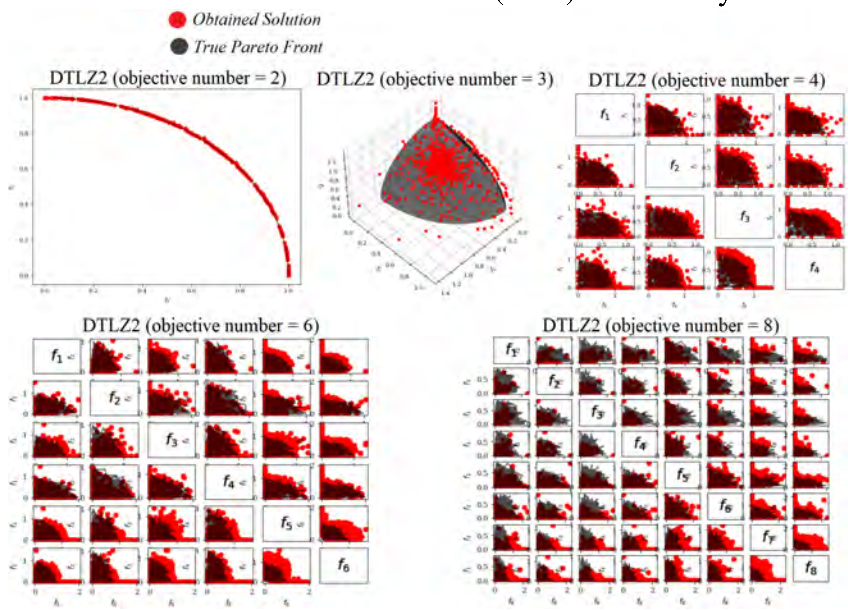


Table 5.12: The number of times each algorithm provided the best average values of the IGD and NHV indicators

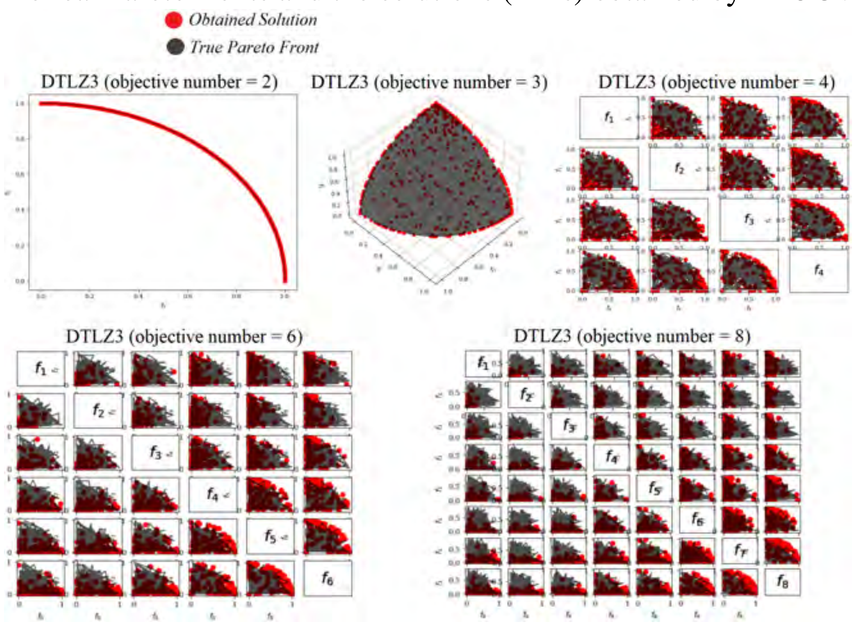
Algorithm		IMOGWO	MOGWO	DMOPSO	MOEA/D-IEps	MOEA/DD	ESPEA
Ranked as first for	IGD	15	0	0	3	3	4
	NHV	11	0	0	1	7	6

Therefore, to check whether the IMOGWO registered a significant improvement over the other optimizers, inferential statistical tests described in the following subsection were applied.

5.3.3 Inferential statistical tests

Moreover, to assess the performance of the various algorithms, statistical tests were carried out on the results of the IGD and NHV indicators. In general, statistical tests are used to determine whether a proposed new algorithm outperforms existing algorithms in solving a given problem. In this context, several statistical comparison procedures have been proposed. [149] presented in detail many possible tests and provided a useful tutorial on the application of these methods in computational intelligence. Thus, since the aim of this work is to compare the proposed algorithm with existing ones, tests based on (1 x N) comparisons were carried out. The Friedman test with Iman-Davenport extension and Holm's post-hoc procedure were chosen as the multiple comparison tests for the performed experiments, as they are recommended by [149]. In fact, the next sections detail the steps taken to apply these two methods.

Figure 5.10: The real Pareto fronts and the solutions (PFs) obtained by IMOGWO for DTLZ3



5.3.3.1 Application of the Friedman test with Iman-Davenport extension

The Friedman test with the Iman-Davenport extension is used to compare one algorithm with a set of other algorithms with the objective of determining whether there is a statistically significant difference between the evaluated methods. To compare IMOGWO to other optimizers, the following steps are taken:

1. The optimizers are ranked for each test problem according to their performance in terms of IGD and NHV, where the best result obtains the lowest rank.

2. The average rank of each optimizer over all test problems is computed to obtain the final rank R_j (Equation 5.1) :

$$R_j = \frac{1}{n}\sum_{i=1}^{n} r_i \tag{5.1}$$

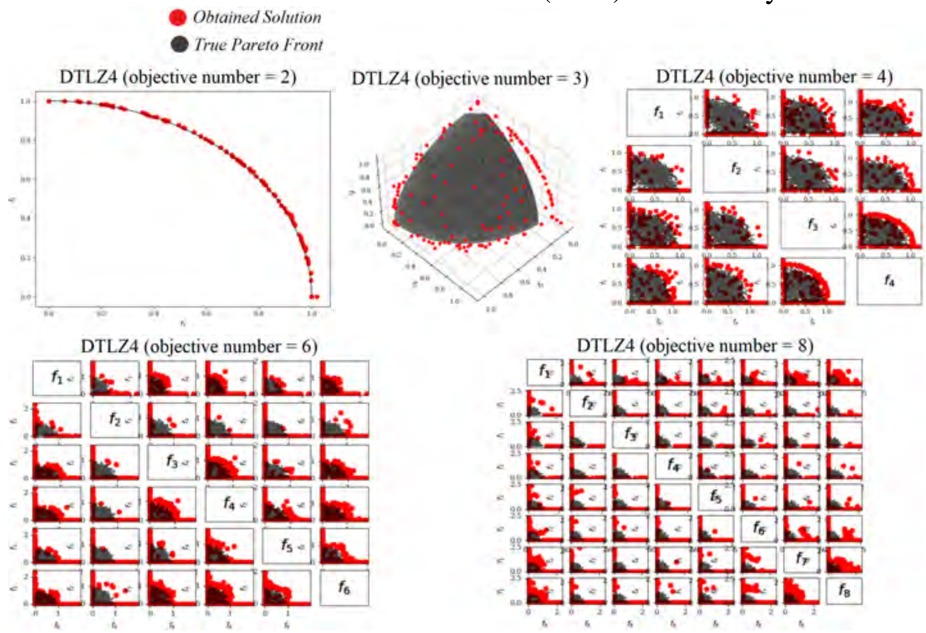
   n is the number of the studied test cases, which is 25. r_i is the rank of the optimizer on the test case i.

3. Friedman's statistics are calculated using Equation 5.2:

$$F_f = \frac{12n}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{5.2}$$

   k represents the number of the compared algorithms, which is 6.

Figure 5.11: The real Pareto fronts and the solutions (PFs) obtained by IMOGWO for DTLZ4



4. The Iman-Davenport's statistics are calculated using Equation 5.3:

$$F_{id} = \frac{(n-1)F_f^2}{n(k-1) - F_f^2} \tag{5.3}$$

5. The corresponding value in the F distribution table is obtained. In our study, this value is equal to $F(5, 120) = 2.29$.

6. $F(5, 120)$ is compared with the obtained statistics. If $F(5, 120)$ is less than F_f and F_{id}, therefore, Friedman's null hypothesis (all algorithms have the same performance) is rejected. Consequently, we can conclude that there is a significant difference between the algorithms compared in terms of the values of the performance indicator.

In fact, Table 5.19 presents the average ranks obtained by the optimizers on all the test problems for the IGD and NHV metrics. The lowest (best) rank is shown in bold, while the next lowest rank is underlined. This table also includes the Friedman and Iman-Davenport statistics, as well as the corresponding values in the F distribution table for the IGD and NHV metrics. For the IGD metric, IMOGWO obtains the best average rank among the compared algorithms. The critical value of the F distribution (5,120), equal to 2.29, is lower than that of Friedman and Iman-Davenport statistics (42.22 and 12.24 respectively). As a result, Friedman's null hypothesis is rejected, indicating that the compared algorithms have significantly different IGD results.

As far as the NHV indicator is concerned, the MOEA/DD has the highest average ranking, closely followed by the IMOGWO. In addition, the Friedman and Iman-Davenport statistics are 30.51

Figure 5.12: The real Pareto fronts and the solutions (PFs) obtained by IMOGWO for DTLZ7

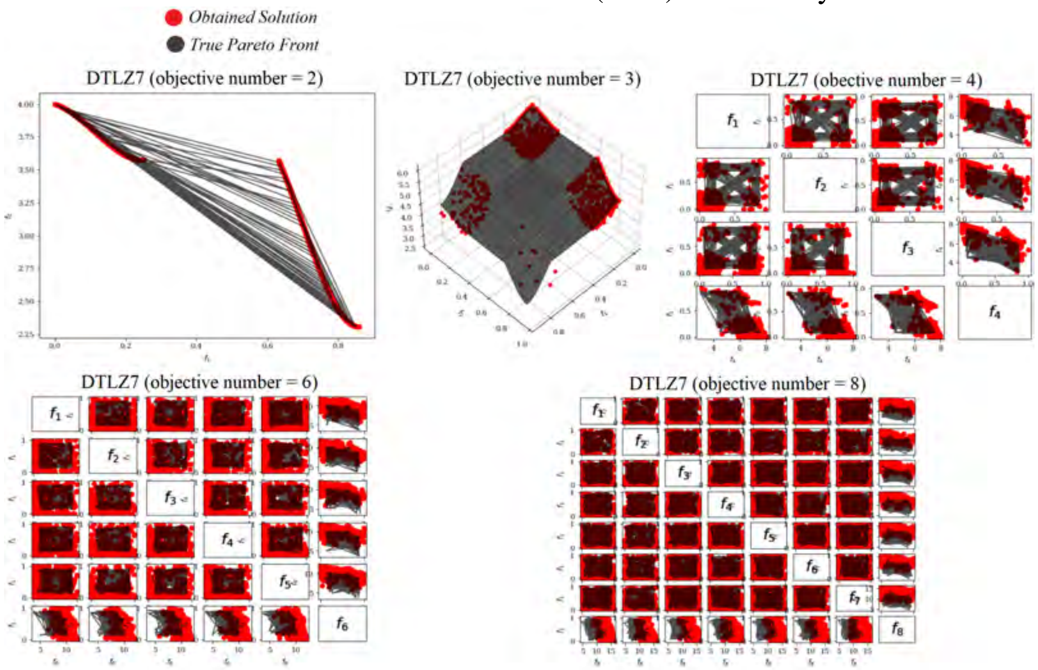
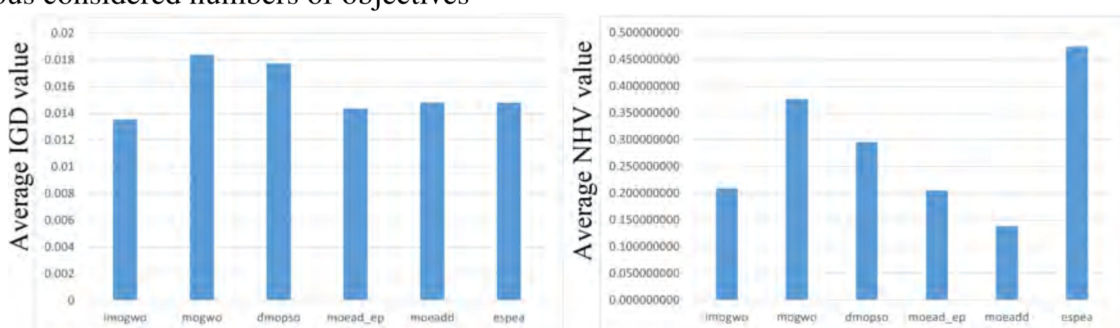


Figure 5.13: Average values of the IGD and NHV indicators for the DTLZ1-4,7 problems and the various considered numbers of objectives



and 7.75, respectively. These statistical values exceed the critical value of the F distribution (2.29), which highlights a substantial difference between the compared algorithms in terms of NHV results. As the results of the Friedman test with Iman-Davenport's extension indicate a significant difference between the studied optimizers in terms of IGD and NHV indicators, we will proceed with the application of the Holm post-hoc test [149].

5.3.3.2 Application of Holm's post-hoc procedure

Holm's post-hoc procedure can be used to compare one algorithm with two or more other algorithms. Moreover, to compare the IMOGWO with other optimizers, this procedure consists in:

Figure 5.14: Performance of the tested optimizers in terms of IGD and NHV indicators for the DTLZ1-4,7 problems and the various considered numbers of objectives

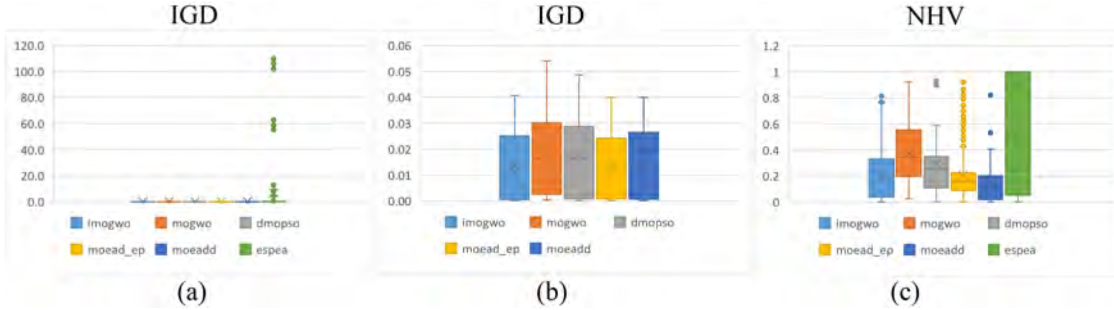


Table 5.13: The average ranks of optimizers and the Friedman and Iman-Davenport statistics for the IGD and NHV metrics

Algorithm		**IMOGWO**	**MOGWO**	**DMOPSO**	**MOEA/D-IEps**	**MOEA/DD**	**ESPEA**
Average Rank	IGD	**1.92**	4.84	4.44	3.02	<u>2.9</u>	3.88
	NHV	<u>2.64</u>	4.88	3.96	3.16	**2.44**	3.92
Friedman statistic for IGD = 42.22 Iman-Davenport statistic for IGD = 12.24 F(5,120) = 2.29							
Friedman statistic for NHV = 30.51 Iman-Davenport statistic for NHV = 7.75 F(5,120) = 2.29							

1. Calculating the z values for the studied optimizers, using Equation 5.4:

$$z_i = (R_{IMOGWO} - R_i)/\sqrt{k * (k+1)/6n} \tag{5.4}$$

   A negative result for z means that the IMOGWO outperforms the other optimizer.

2. Ranking the compared optimizers according to the z values (from the largest to the smallest). r_i represents the rank of optimizer i as a function of the corresponding z_i.

3. Calculating the Holm values for the compared optimizers using Equation 5.5 :

$$Holm_i = 0.05/r_i \tag{5.5}$$

4. Calculating the p values which correspond to the z values. In fact, the p values are based on the normal distribution and Holm's values. If the p value of an optimizer is less than the corresponding Holm value then, the null hypothesis will be rejected. In other words, there is a significant difference between this optimizer and the MOGWO in terms of performance indicator values.

The results of the Holm post-hoc procedure tests, in terms of the IGD and NHV indicators, are presented in Tables 5.14 and 5.15 respectively. We first note that for the IGD indicator (Table 5.14), all values of z found are negative, indicating that Holm's test confirms that IMOGWO outperforms all other optimizers. Moreover, the p-values of MOGWO, DMOPSO and ESPEA are lower than the corresponding Holm values. Therefore, we can conclude that the IGD results of IMOGWO are significantly better than those of MOGWO, DMOPSO and ESPEA. In the same context, although the z-values calculated for MOEA/D-IEps and MOEA/DD are negative, their p-values are higher than their corresponding Holm values. In other words, IMOGWO slightly outperforms MOEA/D-IEps and MOEA/DD in terms of the IGD indicator.

On the other hand, regarding the NHV indicator, the results of the Holm test revealed that the calculated values of z, with the exception of MOEA/DD, are mostly negative,. In other words, for the NHV (Table 5.15), the IMOGWO outperforms the other optimizers. In addition, the p-values of the MOGWO and ESPEA are lower than Holm's values, indicating that the IMOGWO performs significantly better than the MOGWO and the ESPEA. However, the p-values of both the DMOPSO and the MOEA/D-IEps exceed Holm's values, showing that the performance of the IMOGWO is slightly better than that of the DMOPSO and MOEA/D-IEps in terms of the NHV indicator. In summary, the difference between their NHV results is not very marked.

Table 5.14: Results of the Holm post-hoc test for comparing IMOGWO with other optimizers in terms of the IGD indicator

Optimizer	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
MOGWO	-5.5183	5	0.01	0.0001
DMOPSO	-4.7624	4	0.0125	0.0001
MOEA/D-IEps	-2.0788	2	0.025	0.0376
MOEA/DD	-1.8520	1	0.05	0.064
ESPEA	-3.7041	3	0.0167	0.0002

Table 5.15: Results of the Holm post-hoc test for comparing IMOGWO with other optimizers in terms of the NHV indicator

Optimizer	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
MOGWO	-4.2332	5	0.01	0.0001
DMOPSO	-2.4946	4	0.0125	0.0126
MOEA/D-IEps	-0.9827	2	0.025	0.3258
MOEA/DD	0.3780	1	0.05	0.7055
ESPEA	-2.4190	3	0.0167	0.0156

5.3.4 Synthesis

In this section, we use the IGD and NHV metrics to evaluate the performance of the IMOGWO against a set of recently developed and widely used optimizers. The experimental results demonstrate the IMOGWO's capacity to achieve good convergence and efficient distribution. The solutions obtained when solving the tested multi-objective problems satisfactorily converge towards

the true Pareto fronts. This high convergence can be explained by the adjustments made by the IMOGWO to the equations for changing the coordinates of the search agents, thus, improving their proximity to the prey modelled by the MOGWO. In addition, the use of weights to rank the best solutions enabled to improve the convergence towards the final solutions produced by the IMOGWO. This ability to converge is also justified by the updating of the coordinates of the search agents as soon as new positions improve the values of the objectives. In addition, the improved updating of search agent coordinates by the dimension learning-based hunting strategy has strengthened the exploration and distribution of the solutions obtained by the IMOGWO.
In addition, inferential statistical tests were used to compare the IGD and NHV results of the IMOGWO with those of the other studied optimizers. The Friedman test with the Iman-Davenport's extension and the Holm test confirmed that the IGD performance of IMOGWO is significantly better than that of the MOGWO, DMOPSO, MOEA/D-IEps and ESPEA. Although the IGD results of the IMOGWO are slightly lower than those of MOEA/DD, statistical tests have shown that the difference is not significant. Similarly, in terms of the NHV indicator, the IMOGWO significantly outperformed both the MOGWO and the ESPEA, while the differences with the DMOPSO and MOEA/D-IEps were less significant. On the other hand, the MOEA/DD performed the best in terms of NHV.

5.4 Experimental results and discussion of indoor localization using hybrid DL and metaheuristics

In fact, a series of tests has been carried out in order to evaluate the efficiency of the contributions described in the sections 4.5 and 4.6 of the chapter 4: (i) the optimization of the weights of a DL model by a metaheuristic optimization algorithm and (ii) the application of this approach to develop a DL indoor localization model based on UWB ToF measurements. This section first describes the experimental setup and then analyses the obtained results.

5.4.1 Experimental setup

In fact, this section details the carried out experiments, in particular the used dataset, the employed evaluation criteria, the experimental parameters and the implementation choices.

5.4.1.1 The dataset

The data used to apply the proposed solution was generated by the real LocURa4IoT platform [140] [141], which is designed to develop and test indoor localization approaches in the IoT en-

vironments, based on ToF measurements. This platform can also be used to test and evaluate the measurement protocols in the IoT networks.
Moreover, the platform, which is primarily based on the UWB technology, is recognized for its high accuracy, besides, it incorporates the Bluetooth Low Energy (BLE) transceivers and LoRa for certain nodes. To create the dataset, several anchors were deployed in the IoT environment under consideration. A mobile object was moved within this environment to collect measurements relating to the signals exchanged between the object and the anchors.
In addition, the TWR protocol was used to exchange messages between the mobile object (TWR client) and the anchors (TWR servers). The dataset includes various measurements, such as the ToF, the positions of the anchors, the actual positions of the mobile object, the RSSI measurements, the Range measurements, etc. The Range is an estimate of the distance between two objects obtained using a telemetry protocol. This measurement can be estimated from the power of the received signals or by using the ToF. The ToF and the Range measurements were calculated using the TWR protocol [38]. However, the time lag between the TWR server clock and the TWR client clock can lead to inaccurate Range measurements. Therefore, to remedy this problem, the Range measurements in the LocURa4IoT dataset have been optimized, in particular, by taking into account the skew, i.e. the difference in frequency between the two nodes. This dataset can be accessed online at [141].
Therefore, in order to apply the localization solution, the dataset inputs were restructured in order to separate the pairs of inputs and avoid overfitting. However, some elements of the dataset LocURa4IoT, were eliminated, in particular:

- The columns containing unique values, such as object identifiers and the used protocol;
- The columns that do not provide useful information about locating objects.

In addition, the dataset was reorganized so that each row contained information about interactions with the same 5 anchors. The dataset was then divided into three parts: training, testing and validation. In the [150], the authors considered the ratio $(p : \sqrt{p} : (\sqrt{p} + 1))$ as the optimal distribution ratio for these parts of the dataset. Here, the $p = \sqrt{N}$, where the N represents the number of unique rows in the used data.
As a result, the new dataset consists of 3947 unique rows. Therefore, with p =63, the distribution ratio has become $63 : 8 : 9$. In other words, the dataset has been divided as follows:

- 78.75% of the data was assigned to the training set;
- 10% of the data was reserved for the validation set;
- 11.25% of the data was allocated to the test set.

5.4.1.2 Evaluation criteria

Among the available evaluation measures (section 3.3.5 of chapter 3), the following criteria were selected for the experiments: MAE, MSE, R2, Accuracy, Recall and F1-score. Where the MAE is the most frequently used criterion to evaluate the localization solutions. In addition, in [151], it was explicitly recommended that the MSE criterion be used to solve various prediction problems, since it is the simplest, while the others are the most commonly used criteria for evaluating DL models.

5.4.1.3 The experimental parameters

As part of the carried out experiments, Table 5.16 shows the hyper-parameters used to configure the trained and tested DL models.

Table 5.16: Hyperparameters of DL models

Hyperparameter	**Value**
Number of epochs	2000
Batch size	32
Learning rate	0.001
Number of hidden layers	2

5.4.1.4 Implementation

The experiments were carried out on a computer with an Intel i7 processor and 16 gigabytes of RAM. The following choices were made to implement the approach:

- The use of the Python programming language;
- The Jupyterlab 3.5.1 IDE;
- The Keras [152] is a python library devoted to Deep Learning. It provides a high-level interface with high abstraction and simple, consistent Application Programming Interface (API)s, making it easy to develop and test Deep Learning models.
- The implementation of two single-objective optimizers for training the weights of a Deep Learning model. They are based on the following metaheuristic algorithms:
  - GWO [3]: chosen for its recent popularity in various fields [153];
  - AOA [85] : selected for its superior performance to other optimization algorithms and its promising results in solving complex problems [85].

5.4.2 Results

In addition, to evaluate the efficiency of our developed approach, we have carried out a series of tests and compared the created models according to various performance measures. As previously mentioned, we have selected the GWO and the AOA for their recency and efficiency among the existing single-objective metaheuristic optimization algorithms as the best choices for implementing our approach. In addition, the LocURa4IoT dataset offers a variety of measurements for knowing the exact position of mobile objects. Therefore, in order to determine which optimization algorithms and measurement techniques to use in our final solution, we have carried out tests and comparisons, which are presented in the following two subsections.

Firstly, we applied our approach to develop two models; the DL prediction model, using the GWO and the AOA, respectively, and a third model which was developed using the GD optimization algorithm. The results of a comparison between these three models are presented in the following subsection, demonstrating that the model based on our approach and on the GWO offers the best performance.

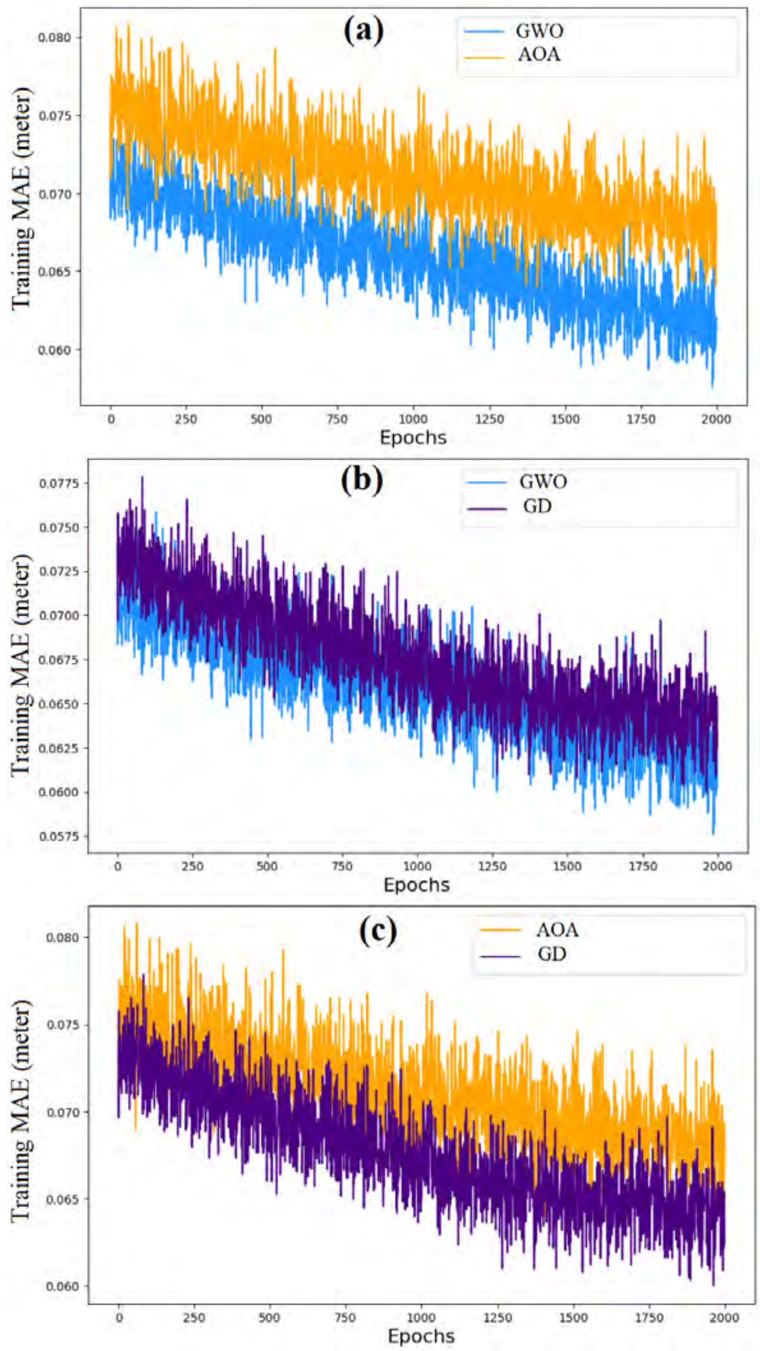
Then, in the second subsection, we developed and compared three other DL models, which are all based on our approach and the GWO, but use different measurement techniques: the ToF, the RSSI and the Range to predict positions. We found that the ToF measurements provide more accurate position predictions than the other techniques. Therefore, our final proposal is to train a DL model using our approach with the GWO and using the ToF measurements. This proposed solution has been compared to three other existing localization solutions among the most recent and commonly used: one based on DNNs, a multilateration solution and an Improved Weighted Centroid Localization Algorithm (IWCL) algorithm. The third subsection provides details of these comparisons and analyses the results to demonstrate the efficiency and the superior performance of our proposal.

5.4.2.1 Comparison of the optimization methods

The performance of the three DL developed models was evaluated using the same hyper-parameters (see Table 5.16). These models are based mainly on the ToF measurements from the LocURa4IoT dataset. This data includes anchor coordinates and ToF measurements. Two models were trained by applying the proposed method (see section 4.5 of the previous chapter) with the GWO and the AOA, respectively. The third model was trained using an existing optimization method (GD).

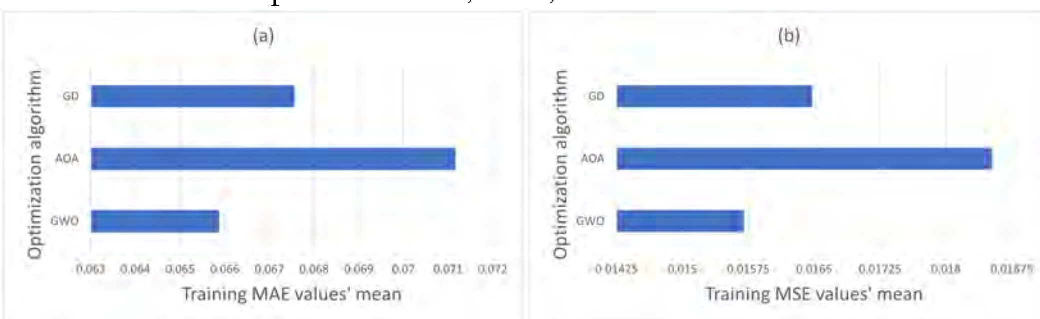
In fact, Figure 5.15 shows the learning curves for the three studied models, illustrating the variation in the MAE values over the training epochs with the three optimizers. Each two curves are plotted together to facilitate the comparison between the models: (i) Figure 5.15(a) compares the two DL models developed with our approach using the GWO and the AOA, (ii) Figure 5.15(b) compares the models developed with the GWO and the GD, respectively, and (iii) Figure 5.15(c) compares the model developed with the AOA to that developed with GD. It is important to notice that throughout the training epochs, the parameters were adjusted to find out the optimal values leading to lower MAEs. This figure shows that optimizing the trainable parameters with our approach, using the GWO, led to the fastest convergence compared to the other two optimizers. In other words, the

Figure 5.15: Learning curves for DL models optimized with the proposed solution (by applying GWO and AOA) and GD



MAE errors were smaller and decreased faster when the GWO equations were used to update the weight values during the learning iterations. Furthermore, this figure indicates that optimization with the AOA metaheuristic is less efficient than that performed with the GWO.
Therefore, from these curves, we can deduce that our approach using the GWO outperforms the other two methods, followed by the GD optimizer. However, the optimization with the AOA turns out to be the least efficient. This conclusion is also confirmed by Figure 5.16, where the mean values of the MAE (Figure 5.16(a)) and MSE (Figure 5.16(b)) errors across all the epochs during the training phase are compared. This analysis confirms that training with the GWO outperforms training with the other two optimizers (the GD and the AOA) and that our GWO-based approach accelerates the convergence to optimal parameters and improves the DL model learning.

Figure 5.16: Comparison of the overall mean values of MAE and MSE errors across all training epochs for the models developed with GWO, AOA, and GD



In addition, the training times for the three models are compared in Table 5.17. In terms of training speed, the optimization with the GD is the most efficient, followed by the GWO and then the AOA. However, this learning time criterion is not essential because it concerns the offline phase (training). During the execution of the IoT applications (online phase), the objects use a pre-trained model to locate themselves.
Thus, based on these results, the optimization using the proposed GWO approach was selected to carry out the remaining tests presented in this section.

Table 5.17: Training time comparison

Optimization algorithm	Training time (seconds)
GWO	328.302
AOA	393.162
GD	281.52

5.4.2.2 Comparison of the measurement techniques

In fact, the initial tests have shown that training based on the approach developed with the GWO gives the best results. Consequently, this optimization method was used to train three DL models with the same hyper-parameters, but using different measurements selected from those proposed by LocURa4IoT. In addition to the anchor positions, the used data include measurements of the ToF, of the Received Signal Strength (RSSI) and Range. The aim of this test phase was to compare the performance of the three developed models in terms of accuracy and position prediction. The trained models were used to predict positions using the test data (11.25% of the dataset). The predicted x and y coordinates, which were obtained from the models, were compared, and these predictions were used to calculate the distance between the actual position of the mobile object and its estimated position. In fact, Figure 5.17 presents a comparative study of the results obtained by the three models. It also shows the error variation curves for x (Figure 5.17(a)) and y (Figure 5.17(b)) coordinates, as well as the distance between the actual and the predicted positions (Figure 5.17(c)). These curves show that the errors obtained from the ToF and the UWB are generally smaller than those obtained from both the RSSI and the Range measurements.

Figure 5.18 confirms that position predictions based on the UWB ToF measurements are more accurate than those obtained with the RSSI and Range. Sub-figures (a), (c) and (e) of Figure 5.18 compare MAE values calculated from predictions for x and y coordinates, as well as measured positions, such as distances between actual and estimated positions. In addition, sub-figures (b), (d) and (f) show comparisons of MSE values for predictions in terms of x and y coordinates as well as the positions.

We also noted that the ToF measurements produced the lowest values of the MAE and MSE criteria. In addition, it is clear that predictions based on Range resulted in smaller MAE and MSE values than RSSI for the x-coordinate and position. However, for the y-coordinate, Range resulted in a lower MAE value and a slightly higher MSE value than RSSI.

Thus, based on these results, we can say that the optimization using the proposed GWO approach and the integration of UWB ToF measurements has been selected as the indoor localization solution proposed by this study, due to its superior performance compared to the other solutions developed and tested. Therefore, it will be compared with other localization solutions in the following subsection.

5.4.2.3 Performance comparison of the solution proposed with the multilateration, the IWCL algorithm and a Deep Learning solution

In fact, the aim of this test section is to compare the performance of our solution Deep Learning-Gray Wolf Optimizer-Time of Flight localization solution (DL-GWO-ToF) with three other commonly used localization methods: a Deep Learning-based method [154], the IWCL [155], and a conventional multilateration method [156]. The aim is to demonstrate that our approach offers superior performance. The position estimation capabilities of each method are evaluated using the same test data.

a) Standard comparisons

The estimation results of the 4 localization solutions are compared in Figures 5.19 and 5.21.

Figure 5.17: Error variation curves for x and y coordinates and the distance between the actual and the predicted positions

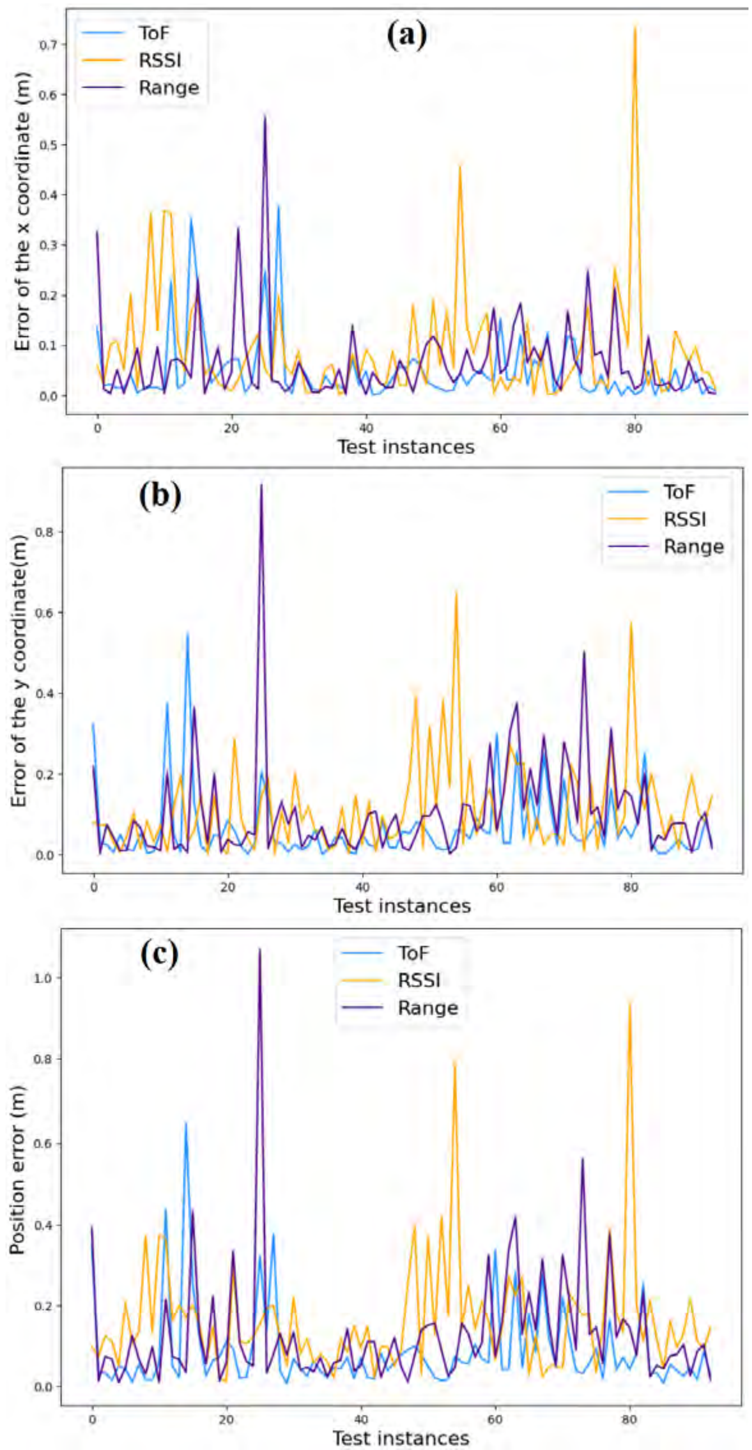
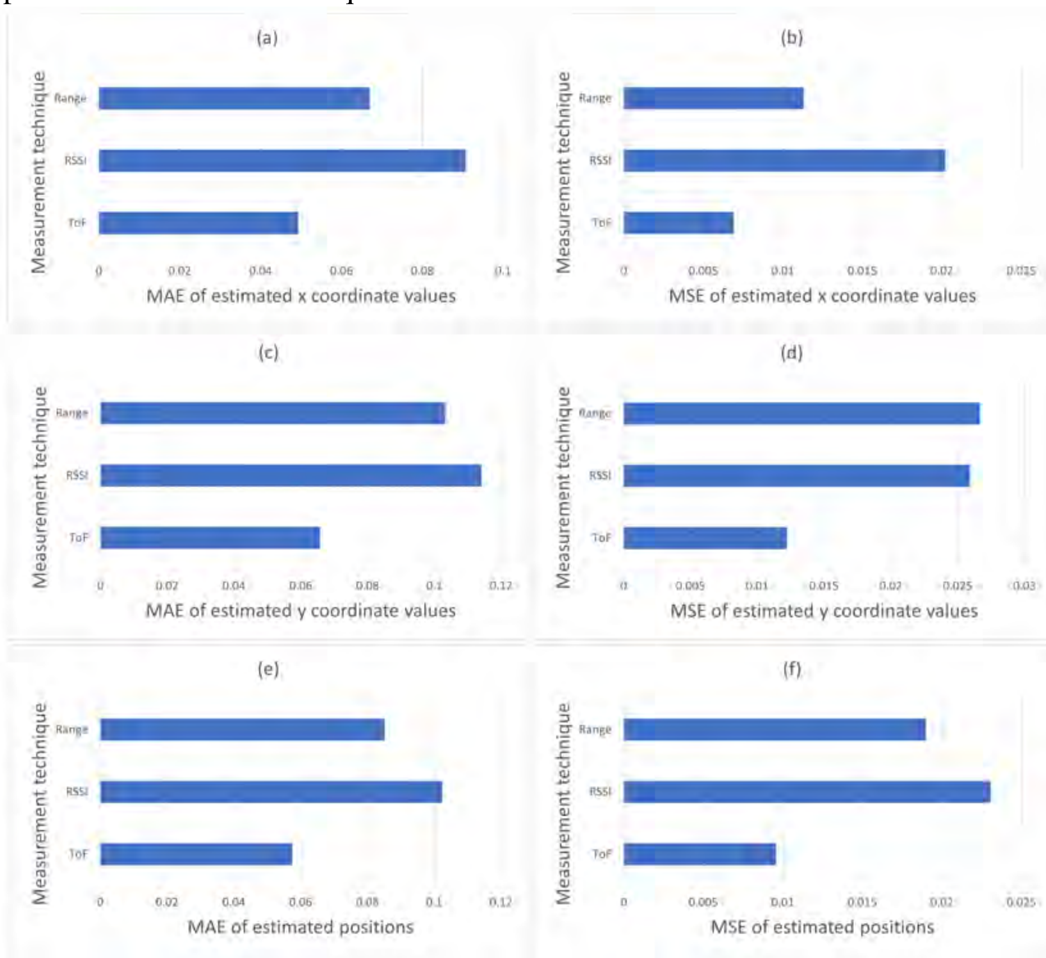


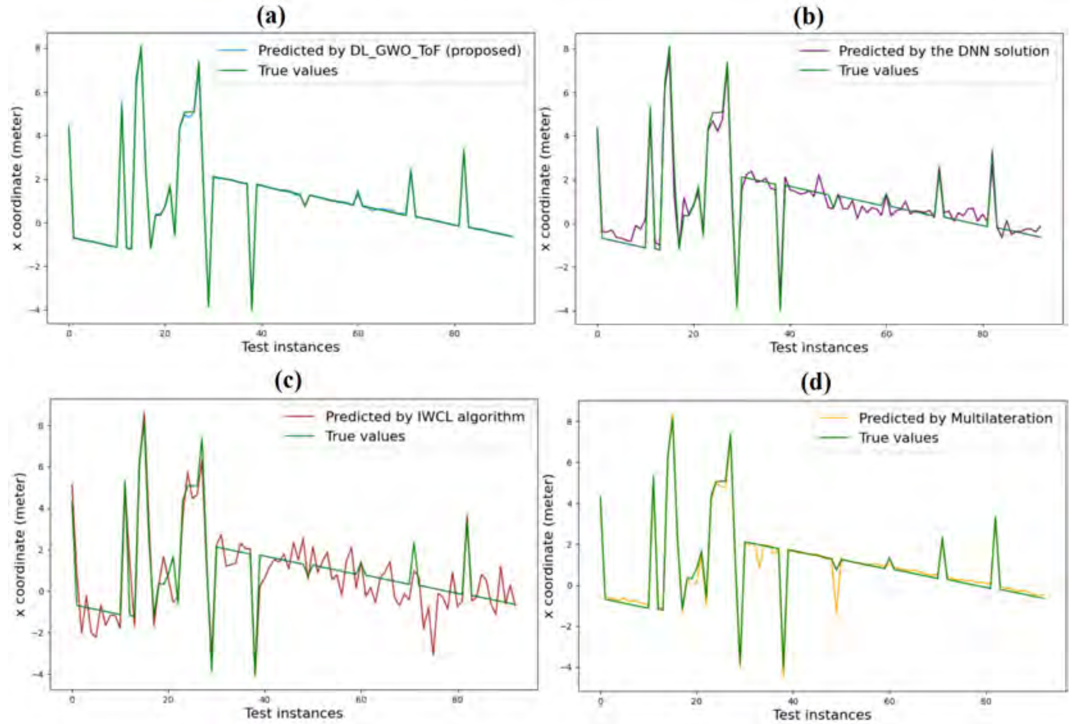
Figure 5.18: Comparison of the overall mean values of MAE and MSE indicators for the predictions of x and y coordinates and the positions (distances between real and estimated positions) for the compared measurement techniques



Figures 5.19 and 5.20 compare the predicted and actual values of the x and y coordinates for the different solutions studied. The curves indicate that the predictions made by the presented approach are more accurate than those of the other methods. In particular, the sub-figures 5.19(a) and 5.20(a) show that the predictions obtained with our method correspond almost perfectly to the actual values of the x and y coordinates. In contrast, the values predicted by the DNN solution are shown in subfigures 5.19(b) and 5.20(b). On the other hand, the closest results in terms of performance are obtained by multilateration (sub-figures 5.19(c) and 5.20(c)). On the other hand, the predictions of the x and y coordinates by the IWCL algorithm, as illustrated in sub-figures 5.19(d) and 5.20(d), are the furthest from the real values.

Figure 5.21 shows a comparative study of the MAE and MSE values calculated from the predictions obtained by the compared solutions. Subfigures 5.21(a) and 5.21(b) represent the MAE and MSE values obtained by predicting the x-coordinate, while the predicted y-coordinate values are

Figure 5.19: Comparison of the real values and the estimated values of the x coordinate obtained by applying the proposed solution, the DNN solution, the IWCL algorithm, and the multilateration method

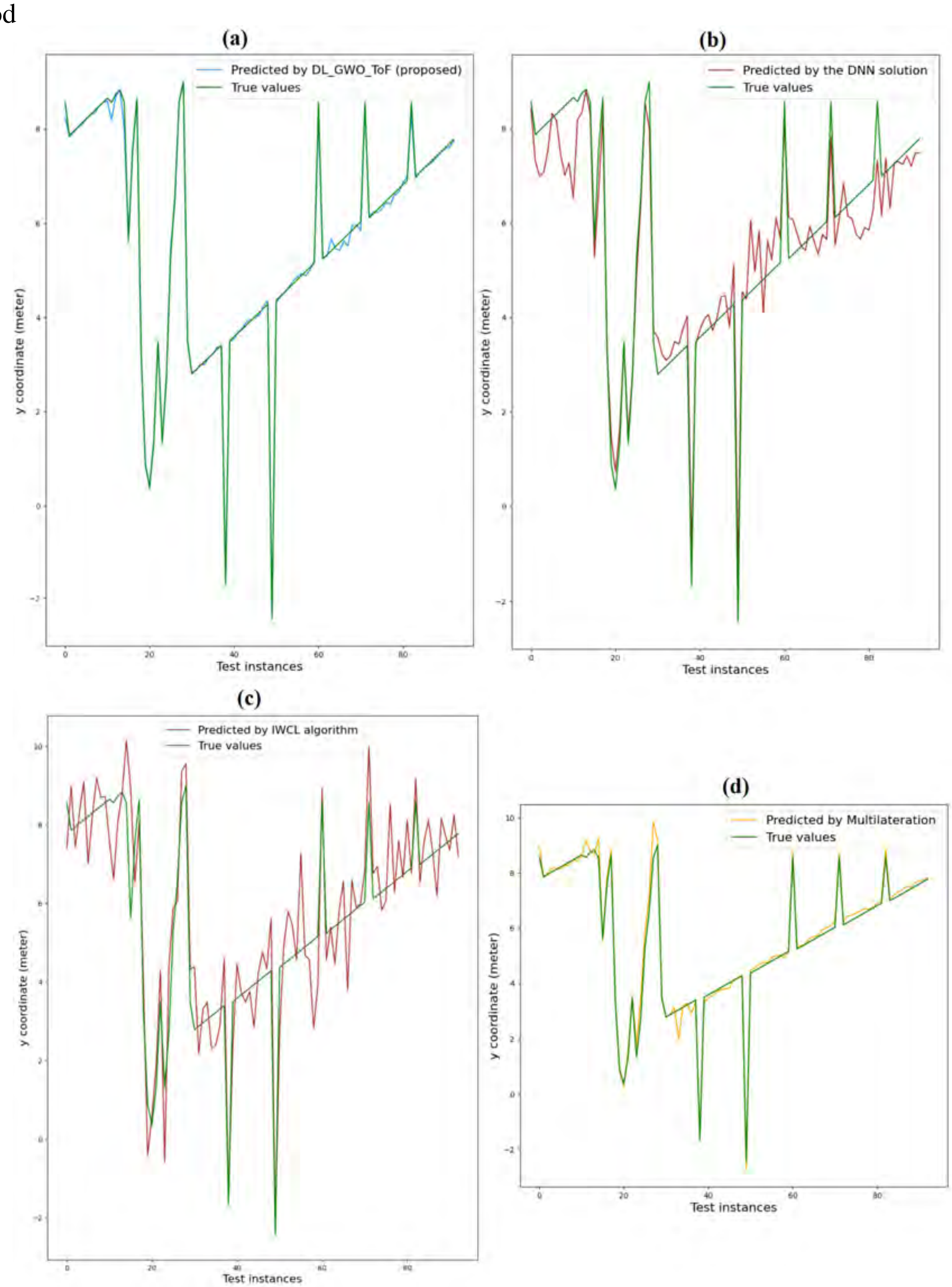


represented in sub-figures 5.21(c) and 5.21(d). In addition, the MAE and MSE values of the distances between the actual and the estimated positions are presented in sub-figures 5.21(e) and 5.21 (f).

In fact, these results show that the developed approach produced smaller MAE and MSE errors for the three considered measurements (x, y and the position). In addition, the comparison of the MAE and the MSE values between the four solutions provides valuable information about their performance. In terms of MAE values for x-coordinate prediction, the proposed solution stands out with the lowest MAE of 0.0492, demonstrating its accuracy in predicting the x-coordinate. The multilateration follows with a higher MAE, where the DNN solution and the IWCL algorithm have considerably higher MAE values, indicating less accurate x-coordinate predictions. This highlights the superior performance of the proposed solution in accurately estimating the x coordinate.

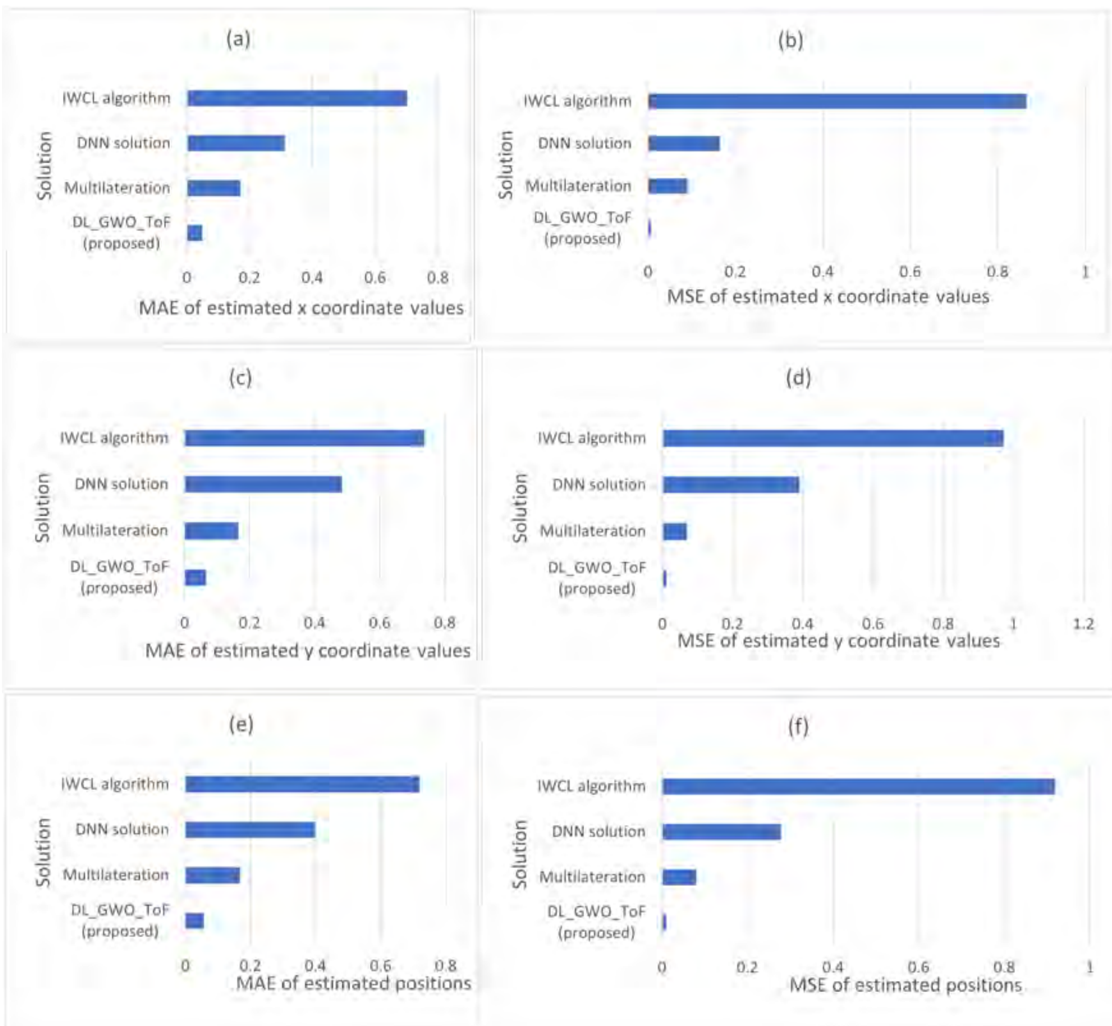
A similar trend is observed in the MAE values for y-coordinate prediction. The proposed solution maintains the lowest MAE, followed by the multilateration, while the DNN solution and the IWCL algorithm exhibit higher MAE values. This highlights the superior accuracy of the proposed approach in predicting the y-coordinate. The overall MAE values for position prediction highlight that the proposal outperforms the other solutions, as it provides the most accurate position estimates. According to the MSE values, the proposal outperforms the other solutions, particularly in terms of x and y coordinate predictions. In fact, the rate at which the proposed localisation solution achieved the minimum distance between actual and estimated positions was 93.54%. These

Figure 5.20: Comparison of the real values and the estimated values of the y coordinate obtained by applying the proposed solution, the DNN solution, the IWCL algorithm, and the multilateration method



results collectively demonstrate the efficiency of the proposed solution in achieving accurate indoor localisation, which is crucial for the IoT applications.
In addition, the presented approach provided an average error between the actual and the predicted positions of 0.057m. These results also demonstrate that the proposed method outperformed recent localization solutions, described in section 2.4.2, where the best average distance error was a few tens of centimeters.
In addition, Table 5.18 provides a comprehensive analysis of different localisation solutions, evaluated according to several key performance criteria: localisation accuracy, mean position error, coefficient of determination R2, recall, F1 score, as well as the times required to apply these methods.

Figure 5.21: Comparison of the overall mean values of MAE and MSE errors for the predictions of x and y coordinates and the positions (distances between real and estimated positions) for the compared solutions



In addition to the MAE and MSE, the determination coefficient R2 can generally assess the model performance for prediction problems. However, precision, recall and F1 scores are measures used for classification problems, involving the classification of data points into categories. In the case of regression, our objective is rather to predict a continuous numerical value, and not categorize the data into classes. For this reason, a customised version of these metrics is developed to evaluate the localisation problem. In practice, precision, recall and F1 score are calculated for regression by considering values within a tolerance margin as true positives. Since the choice of this margin can have a significant impact on the results, its value is generally defined according to the requirements of the application for which the localisation solution is intended. In our study, we use a threshold of 0.5 meters as an example. In other words, a prediction is considered successful only if the difference between the actual and predicted values is less than 0.5 metres. Therefore, in this context, precision, recall and F1 score are used to measure the extent to which solutions correctly identify positions. Furthermore, in Table 5.18, the column "training time" indicates the time required to train the localisation models. This measure is not applicable for the IWCL and multilateration solutions. The "estimation time" column reflects the time required to make predictions or estimate the positions.

Our proposed solution outperformed other methods by achieving a localization accuracy of 98.92%, while the multilateration method achieved an accuracy of 89.24% (with a threshold of 0.5 metre). In fact, among the solutions compared in Table 5.18, the proposed approach stands out as the most accurate and reliable as it achieved a remarkable localization accuracy of 98.92% with an average position error of just 0.057 meter. In addition, it has an R2 score of 0.998, indicating a high position predictability. Moreover, this method excels in terms of recall (0.978) and F1 score (0.996), while demonstrating its capacity of correctly identifying positions in the vast majority of cases. However, it requires a longer training time of 328.302 seconds, although it can offer a fast estimate in just 0.155 second.

In contrast, the IWCL algorithm had the lowest localization accuracy, at 19.35%, with a high mean position error of 0.719 meters. In addition, it has a limited R2 score of 0.817, suggesting a reduced predictability. Its recall score (0.120) and F1 score (0.562) are also the lowest, indicating the existence of some difficulties in accurately identifying positions. However, its estimation time is fast at 0.064 second, making it potentially suitable for real-time applications.

The multilateration method provides a relatively fast estimate in 1.982 seconds with a precision of 89.24% and a mean position error of 0.167 metres. It has a good recall (0.805) and a good F1 score (0.969), indicating its ability to correctly identify positions with a certain compromise between precision and recall.

Table 5.18: Comparison of the proposed solution and other studied localization methods

Solution	Localization Accuracy	Mean Position Error (meter)	R2	Recall	F1-score	Training Time (sec)	Estimation Time (sec)
DL-GWO-ToF (proposed)	98.92%	0.057	0.998	0.978	0.996	328.302	0.155
Multilateration	89.24%	0.167	0.983	0.805	0.969	-	1.982
DNN Solution	51.61%	0.397	0.942	0.377	0.829	335.13	0.177
IWCL Algorithm	19.35%	0.719	0.817	0.120	0.562	-	0.064

Finally, the solution based on neural networks (DNN) has a localisation accuracy of less than 51.61%, with relatively lower recall scores (0.377) and F1 scores (0.829). Although its accuracy is moderate, it offers training and estimation times comparable to the proposed solution.
Furthermore, the time required to predict the position using the introduced approach was 0.155 seconds, while the multilateration solution required 1.982 seconds to estimate the coordinates. Although the developed solution outperformed the other method in terms of estimation time, it is important to note that the proposed DL solution required a preliminary offline phase, taking 328.302 seconds to train the prediction model. On the other hand, the multilateration method did not require any additional steps apart from real-time position estimation.

b) Statistical inferential tests

Therefore, to assess whether the proposed solution represents a significant improvement over the other localization methods, we used inferential statistical tests, as described in this subsection. In fact, statistical tests are commonly used to evaluate the relative performance of a new algorithm compared to existing solutions for a specific problem. Therefore, since our objective is to evaluate the proposed solution against existing alternatives, we performed tests involving (1 x N) comparisons. For this purpose, we used the Friedman test with the Iman-Davenport extension and Holm's post-hoc procedure, following the recommendations of [149]. The steps of implementing these two methods are detailed in the following paragraphs.

- **Application of Friedman's test with Iman-Davenport's extension** :

  The Friedman test, with the Iman-Davenport extension, is used to compare an algorithm with a collection of other algorithms and statistically identify significant differences between them. In this context, it is used to evaluate the proposed solution against other solutions, in terms of position errors (distance between actual and predicted positions). The steps involved in applying this test are detailed in section 5.3.3.1.

  Table 5.19 shows the average ranks obtained by the compared localization solutions for all test instances in terms of the position error metric. The most favorable rank (indicating superior performance) is highlighted in bold, and the next most favorable rank is underlined.

Table 5.19: Average ranks of localization solutions and calculation of Friedman and Iman-Davenport statistics for position errors

Solution	DL-GWO-ToF (proposed)	Multilateration	DNN Solution	IWCL Algorithm
Average Rank	**1.45**	2.77	5.41	9.88
Friedman statistic = 9.53 Iman-Davenport statistic = 44.53 F(3,276) = 2.08				

In addition, Table 5.19 provides the statistical results of the Friedman and Iman-Davenport tests, as well as their associated values in the F distribution table.

The analysis of the provided results showed the existence of significant differences in the average performance ranks among the compared localization solutions. In fact, the proposed solution obtained the best average rank of 1.45, demonstrating its superior position prediction accuracy. The multilateration method also demonstrated strong performance with an average rank of 2.77. In contrast, the DNN-based solution and the IWCL algorithm have higher average ranks of 5.41 and 9.88, respectively. These results indicate that the proposed DL-GWO-ToF approach outperforms the other alternatives, and that multilateration also offers reliable performance.

The Friedman and Iman-Davenports statistics provide additional information on the significance of these differences. In fact, Friedman's statistics, with a value of 9.53, show that there are significant variations in the performance ranks of the solutions. This corresponds to the differences observed in the average ranks. The Iman-Davenport statistics, which is equal to 44.53, completes the Friedman test by highlighting the disparities between the solutions. Furthermore, when compared to the critical value of the F distribution table (F(3.276) = 2.08), which serves as a threshold, the Friedman obtained statistics of 9.53 exceed this threshold. Therefore, this result supports the conclusion that there is a statistically significant distinction in performance between the localization solutions.

- **Application of Holm's post-hoc procedure** : The steps involved in applying Holm's post-hoc procedure are detailed in 5.3.3.2.

Table 5.20: Results of Holm's post-hoc test for comparison between localization solutions in terms of position errors

Solution	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
DL-GWO-ToF	0	4	0.0125	1.0
Multilateration	-2.57	1	0.05	0.0102
DNN Solution	-7.44	2	0.025	< 0.0001
IWCL Algorithm	-14.62	3	0.0167	< 0.0001

Table 5.20 summarizes the results of Holm's post-hoc procedure applied to the position error indicator. It is notable that all the obtained z-values are negative, confirming that the pro-

posed solution (DL-GWO-ToF) consistently outperforms all the other compared localization solutions.

The comparative analysis of the localization solutions using the Holm's post-hoc test highlights the superior performance of the DL-GWO-ToF solution. DL-GWO-ToF, which is considered the benchmark against which the other solutions are evaluated. Moreover, its p-value of 0.0125, corrected by Holm, confirms its superiority and shows that the DL-GWO-ToF solution outperforms all the other studied methods.

On the other hand, the other solutions, the multilateration, the DNN solution and the IWCL algorithm, show significantly poorer performance compared with DL-GWO-ToF. Their negative z-values, particularly those of the DNN solution and the IWCL algorithm, with -7.44 and -14.62 respectively, underline their significantly poorer performance. These results, combined with their lower ranks and very low Holm-corrected p-values, indicate that these solutions perform significantly worse than DL-GWO-ToF. Overall, the inferential statistical analyses confirm the superiority of the DL-GWO-ToF solution over the other localization evaluated methods.

5.4.2.4 Synthesis

In fact, several conclusions emerged from the carried out tests and the obtained results:

- Using the GWO-based optimization to train the parameters of the DL model, as proposed in this work, we found a superior performance compared to the other two optimization methods (AOA and GD). More precisely, the GWO approach accelerates the convergence towards the optimal parameters and improves the learning process of the DL model.
- The positions predicted using UWB ToF measurements are more accurate than those obtained using RSSI and Range.
- The proposed localization solution has an average error of only 0.057 meters between actual and predicted positions, with a localization accuracy of 98.92%. It manages to estimate positions consistently and accurately, outperforming the other localization solutions tested (notably those based on DNN, the IWCL algorithm and the multilateration method).
- This proposed approach outperforms recent indoor localization solutions, whose best results achieve an average error of a few tens of centimeters.
- Although the proposed approach provides a quick estimation compared to other solutions, it requires longer training time. Reducing the training time could be a feasible perspective to enhance this approach. Moreover, retraining the localization model is necessary in case of

evolutions or changes in the network topology (number and positions of anchors). Therefore, the development of a DL model capable of self-learning and self-adapting to these changes proves to be another perspective to consider.

This contribution and its results are the subject of our publication [142].

5.5 Experimental results of IMOGWO- based geographic routing

This section is devoted to the presentation and analysis of the results of the experiments carried out to test the contribution of the geographic routing described in section 4.7. The main objective of these experiments is threefold: to evaluate the performance of the proposed geographic and hybrid routing solution, to compare this performance with that of other existing routing solutions, and to test the effectiveness of the IMOGWO approach presented in section 4.4, by applying it to a real problem other than the DTLZ reference problems. These analyses and comparisons will make it possible to understand the advantages and the limitations of the proposed method in a practical context.

5.5.1 Experiments using hybrid simulations

In fact, we have implemented hybrid simulations, integrating both simulated and real elements. The integration of real objects in these simulations makes it possible to get as close as possible to real conditions and environments. This hybrid approach makes it possible to obtain results that are closer to reality while retaining the flexibility and accuracy of virtual simulations.

5.5.1.1 The IoT simulator

We have chosen to use the CupCarbon IoT simulator [157] [158] to carry out the hybrid simulations. CupCarbon is a tool widely used in research for its ability to model and simulate networks of connected objects. Its main advantage lies in its ability to integrate both real and simulated objects, each with a variety of characteristics, into a hybrid simulation. The simulated objects are coded using the Python language.

5.5.1.2 The M5StickC real objects

As part of our hybrid simulations, we have chosen to integrate the real objects M5StickC [143]. They offer a multitude of features for real-time data collection and are widely used in various areas of the IoT. Their compact size and versatility make them an ideal choice for our experiments, enabling seamless integration into our hybrid simulations. The real objects and the simulated ones communicate using the Message Queuing Telemetry Transport (MQTT) protocol [159].

Figure 5.22: The mini-ESP32 M5StickC IoT devices used in the experiments

5.5.1.3 The experimental setup parameters

Table 5.21 shows the parameters used in the experiments. The number of executed cycles is 500. In fact, a cycle is an end-to-end communication that occurs between 2 network objects to send a message, thus requiring the establishment of a multi-hop route to exchange data. Moreover, in terms of the number of objects, we have included a total of 100, 90 of which are simulated: 70 fixed and 20 mobile objects, while 10 are real M5StickC devices. The mobile objects follow random trajectories. As far as the initial energy is concerned, the simulated objects have a variable initial energy, since our aim is to consider the objects that are heterogeneous in terms of energy capacity. In contrast, the real M5StickC devices are equipped with 95 mAh batteries at 3.7V. The energies for transmitting a bit, (E_{Tx_bit}), and receiving a bit, (E_{Rx_bit}), are also given in the table. The used communication technology is Wi-Fi. Such network is assumed to be in ad-hoc mode, thus direct communication between the various nodes without employing centralized infrastructure. Figure 5.23 represents the proposed network architecture.
Figure 5.24 illustrates the model of the hybrid simulation carried out with CupCarbon, where the communication links are not displayed. The objects are randomly distributed in the network. On the other hand, Figure 5.25, shows the same model with the displayed communication links.

5.5.2 The evaluation criteria

In this context, we have selected evaluation criteria to assess our proposed approach and compare it with other existing solutions. The used performance indicators are:

- The transmission latency: it is the time spend between the transmission of a data packet by the initial source and its reception by the final destination.

Figure 5.23: Proposed network architecture

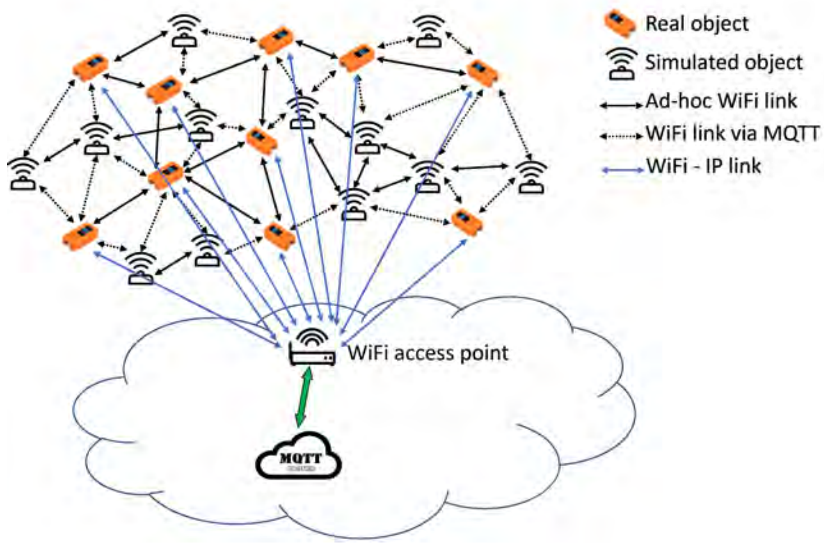


Table 5.21: Parameters of the experiments

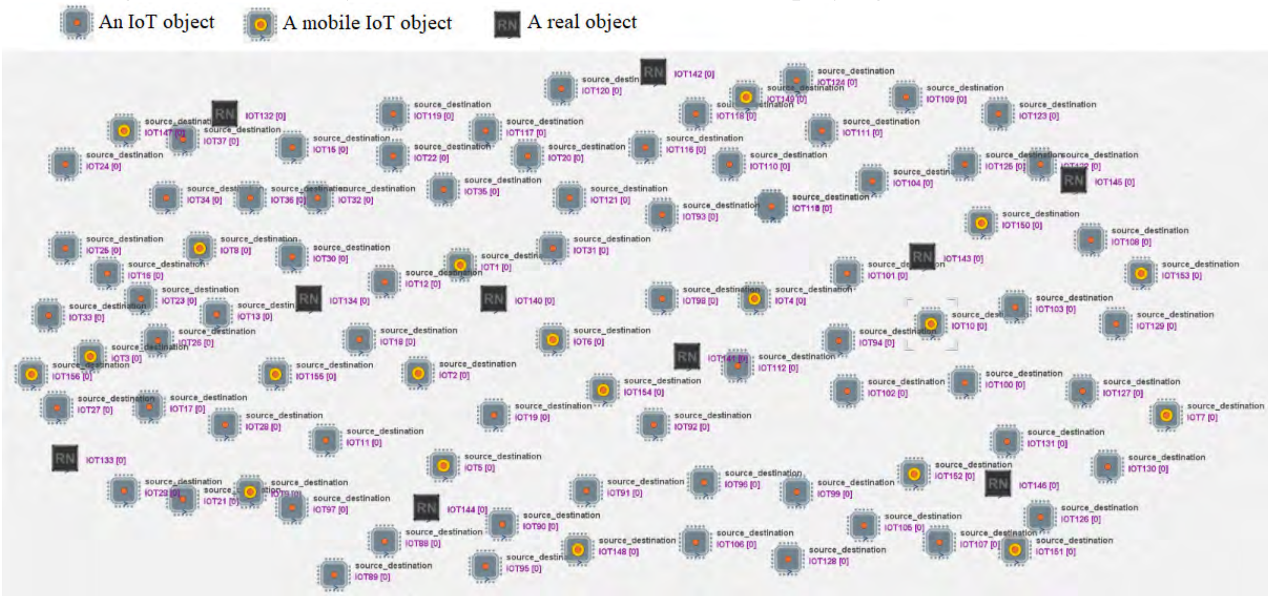
Parameter	Value	
Number of cycles	500	
Number of objects	total	100
	simulated fixed	70
	simulated mobile	20
	real M5StickC	10
Mobility	random	
Initial message size	64-80 bits	
Initial energy	simulated objects	variable
	real M5StickC objects	95 mAh at 3.7V
E_{Tx}	simulated objects	10 nJ/bit
E_{Rx}	simulated objects	5 nJ/bit

- The PDR: it designates the proportion of the sent data packets that reach their destination successfully (equation 5.6). It refers to the ability of the routing protocol to correctly send data packets to their destination without loss or corruption.

$$PDR = \frac{number\_of\_packets\_received\_correctly}{total\_number\_of\_packets\_sent} \times 100\% \tag{5.6}$$

- The number of active objects during the executed cycles.
- The First Node Dies (FND): it is the period between the beginning of communications within the network and the time when the first node completely exhausts its energy and stops operating. It refers to the period of network stability.

Figure 5.24: The hybrid simulation model without displaying communication links



- The control overhead: It is the amount of additional control data used by routing control messages. Low control overhead reduces the impact on network performance. The control overhead can be calculated using the following equation:

$$Control\_overhead = \frac{Total\_size\_of\_control\_messages}{Total\_size\_of\_transmitted\_data\_messages} \times 100\% \qquad (5.7)$$

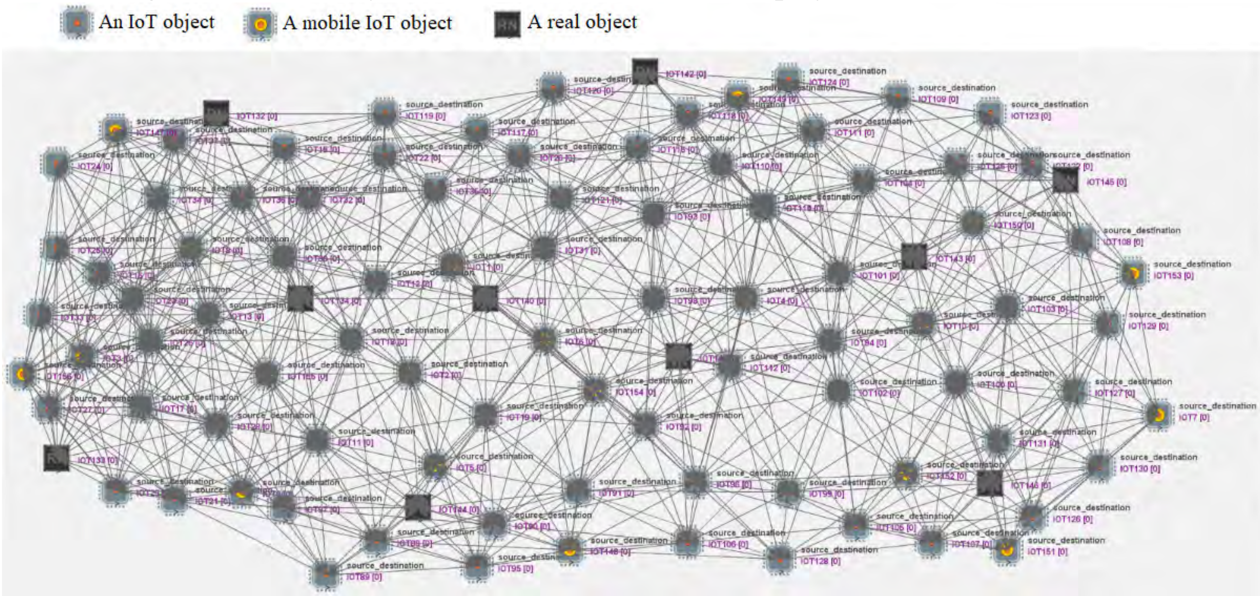
We have actually chosen these performance criteria because of their wide use and ability to evaluate the various performance aspects of the studied routing solutions, including their efficiency, reliability, energy optimization, load balancing and stability.

5.5.3 The comparative routing algorithms

In fact, in the comparative analysis, we compared the results of the following routing solutions:

- Our geographic and hybrid routing approach described in section 4.7 developed with IMOGWO (section 4.4);
- Our geographic and hybrid routing approach described in section 4.7 developed with MOGWO [2];
- QoS-based routing proposed and presented in [83];
- Geographic routing proposed and presented in [109].

Figure 5.25: The hybrid simulation model with display of communication links



Firstly, our geographic and hybrid routing approach has been developed in two versions, applying IMOGWO and MOGWO, respectively. Therefore, since the simulator can provide the positions of objects, the positions considered during geographic routing are calculated from the provided positions, while taking into account the average error obtained by our localization solution proposed in the previous section, which is 0.057 meters. The aim of comparing the results obtained by the two optimization algorithms is to demonstrate the ability of our IMOGWO proposal to produce better results than the MOGWO in solving a real routing problem, apart from the DTLZ reference problems. Secondly, we selected the two proposed routing solutions, respectively in [83] and [109] in order to compare the results obtained with our approach. In fact, this choice is justified by the fact that they are among the routing solutions, based on metaheuristic optimization, recently proposed (see Table 3.3) and that they have demonstrated promising performance. The routing algorithm described in [83] is based on AcNSGA-III, a proposed metaheuristic optimization algorithm. The evaluations carried out to test it used M5StickC objects. The routing algorithm described in [109] is of the same geographical nature as our proposal. It is based on the BFO metaheuristic optimization algorithm.

The compared routing algorithms are all evaluated under the same conditions and the same initial environment. Table 5.22 contains the parameter values of the meta-heuristics, AcNSGA-III and BFO, applied to develop these algorithms.

Table 5.22: Parameters of the AcNSGA-III and BFO algorithms

Parameter	**Value**	
Population size	100	
Number of generations	500	
AcNSGA-III	Crossover index	0.8
	Crossover operator	Simulated Binary Crossover
	Crossover probability	20
	Mutation index	50
	Mutation operator	Bit-flip
	Mutation probability	1/450
BFO	Swim steps	4
	Chemotactic steps	100
	Reproduction steps	5
	Elimination and dispersal steps	2
	Elimination probability	Bit-flip
	Run-length unit	$10^{-3} \times R$

5.5.4 Results and discussion

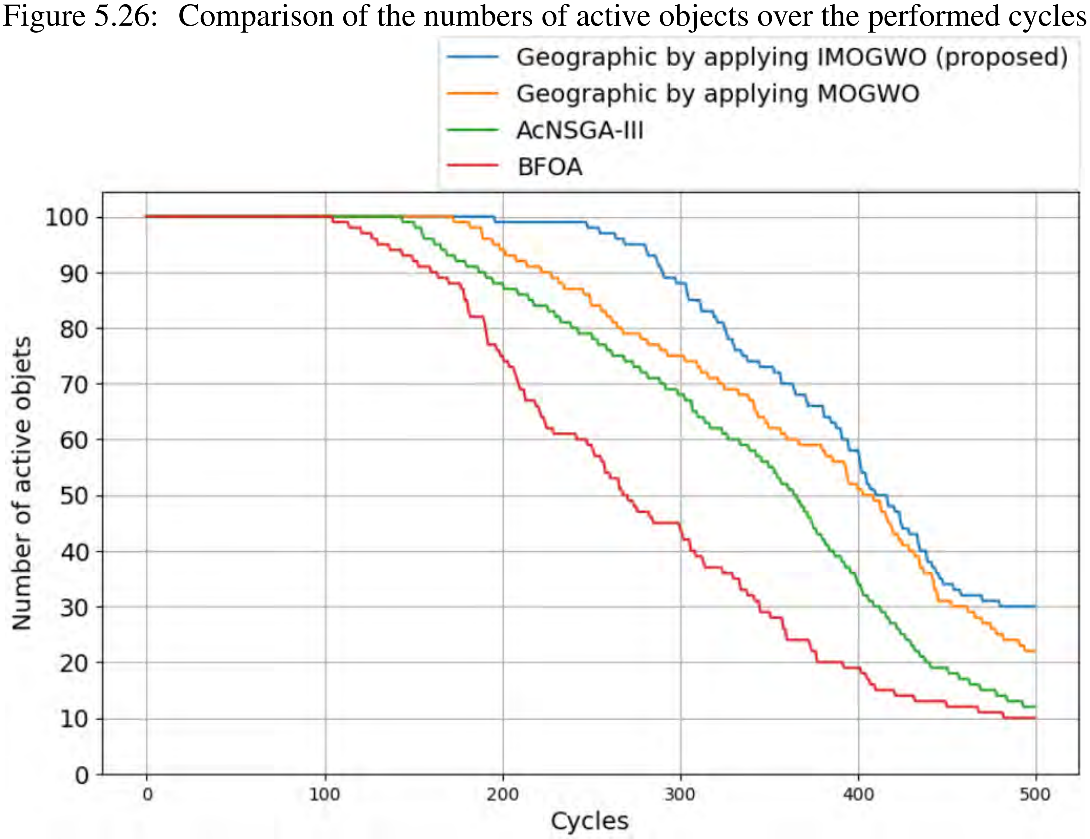
The results of the routing algorithms are compared using two approaches: standard comparisons and inferential statistical tests. Standard comparisons allow a qualitative assessment of the performance of the different algorithms by directly examining the results obtained. In addition, we used inferential statistical tests for a more in-depth analysis to determine whether the differences observed between the performance of the algorithms are statistically significant.

5.5.4.1 Standard comparisons

In fact, Figure 5.26 illustrates the variation in the number of active objects over the performed cycles for the compared routing algorithms. Actually, an object becomes inactive when it exhausts its energy, which is represented by a decrease in the number of active objects over the cycles. The results show that our geographic routing approach with IMOGWO and that with MOGWO are distinguished by their ability to maintain a relatively constant number of active objects over a large part of the simulation cycles, suggesting a good initial distribution of the energy load. However, the IMOGWO approach seems to offer better long-term stability, with a gradual decrease in the number of active objects from a certain point, while the MOGWO shows a tendency towards a more rapid decrease. On the other hand, although the AcNSGA-III and the Bacteria Foraging Optimization Algorithm (BFOA) algorithms start with an initial energy load distribution similar to the geographic routing approaches, they appear to be less efficient in managing energy consumption, with a more rapid decrease in the number of active objects over time.
In fact, these results are confirmed by Figure 5.27, which compares the round the FND values for the routing algorithms. FND refers to the point at which the first object fully exhausts its energy. In other words, Figure 5.27 compares the stability periods offered by the studied routing solutions.

Thus, according to these results, it seems clear that the geographic routing with IMOGWO and MOGWO demonstrates a superior performance compared to AcNSGA-III and BFOA, providing better management of energy consumption and better stability in load balancing between the network nodes. In addition, the use of the IMOGWO algorithm in geographic routing outperforms that of the MOGWO, providing a more balanced distribution of the energy load, which translates into a superior ability to efficiently manage the multi-objective routing problem.
Figure 5.28 compares the values of the average transmission latency obtained by the routing algorithms during the performed cycles. In fact, each value represented in the figure corresponds to the average latency evaluated over a period of 100 successive cycles. Figure 5.29 also compares the average transmission latency values for all the performed cycles. Consequently, the results represented by the two figures reveal significant differences in the transmission latencies between the different evaluated routing algorithms. For our geographic routing approach using the IMOGWO, a slight variation in average latencies during the cycles is observed, with a total average of 1.93 seconds. In contrast, the approach using the MOGWO shows slightly lower average latencies, with a total average of 1.78 seconds. Then, the AcNSGA-III and BFOA algorithms show higher average latencies, with total values of 2.57 and 2.36 seconds, respectively. In conclusion, the results indicate that our proposed geographic routing with the IMOGWO and the MOGWO offers a superior

Figure 5.26: Comparison of the numbers of active objects over the performed cycles



performance in terms of transmission latency compared to the AcNSGA-III and BFOA algorithms. Moreover, between the two geographic routing approaches, the one using the MOGWO seems slightly more efficient in reducing the total average latency. This difference could be attributed to the fact that IMOGWO involves more complex calculations than MOGWO, which allows it to better optimize transmission paths. However, this may lead to a slight increase in latency compared to the MOGWO.

Figure 5.30 illustrates a comparison between the routing algorithms evaluated in terms of PDR values over the cycles performed. Each value shown in the figure corresponds to the PDR evaluated over a period of 100 successive cycles. On the other hand, the represented PDR values highlight a better reliability provided by our proposal compared to the other studied routing algorithms. For our geographic routing approach applied with both the IMOGWO and the MOGWO, the PDR is relatively high, with values varying between 84% and 97%. In contrast, the AcNSGA-III and BFOA algorithms have lower PDRs, varying between 79% and 92%. These observations are confirmed by the total PDR values calculated for all the cycles, as shown in Figure 5.31. Our geographic routing proposal with the IMOGWO offers the highest total PDR (92.2%), followed by our proposal with the MOGWO (90.6%). In contrast, the AcNSGA-III and BFOA algorithms perform worse, with total PDRs of 86.6% and 85%, respectively. In conclusion, the results show that our geographical routing proposal applied with IMOGWO outperforms the other algorithms evaluated in terms of the reliability of packet transmission in the network. However, a decrease in the PDR across the cycles for all the compared algorithms can be seen. This can be explained by the fact that the number of

Figure 5.27: Comparison of the values of round the FND

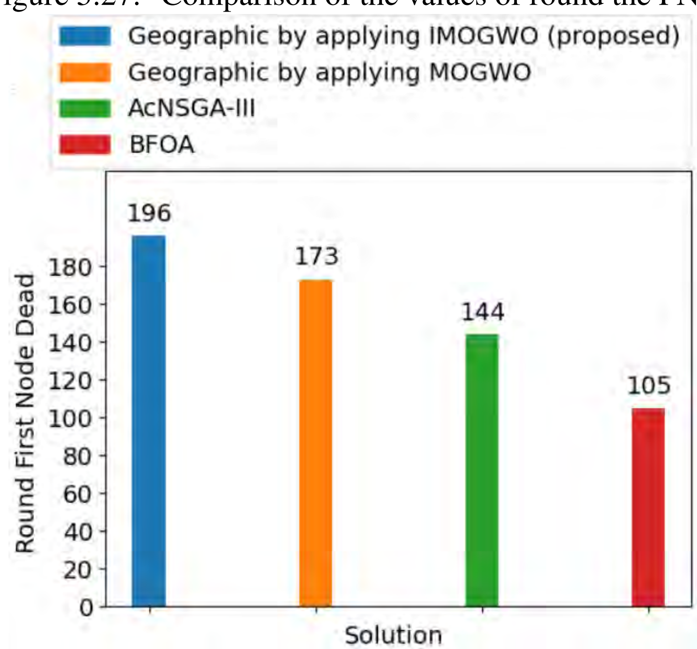


Figure 5.28: Comparison of the values of the average transmission latency during the performed cycles

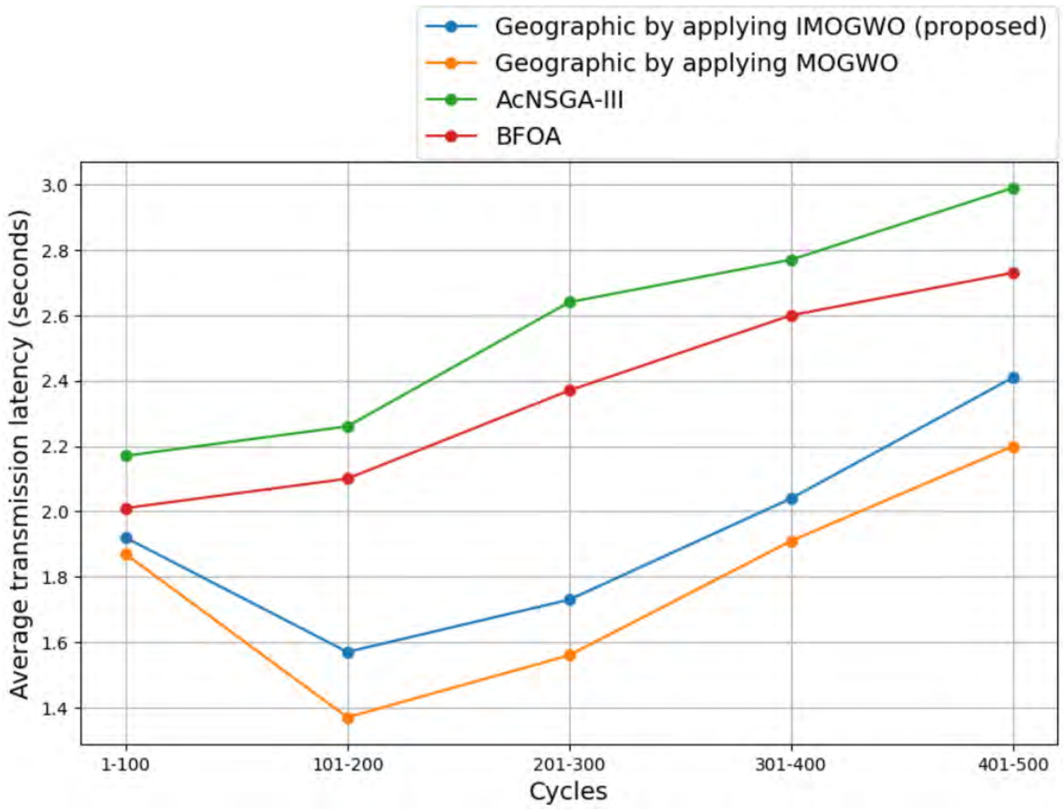
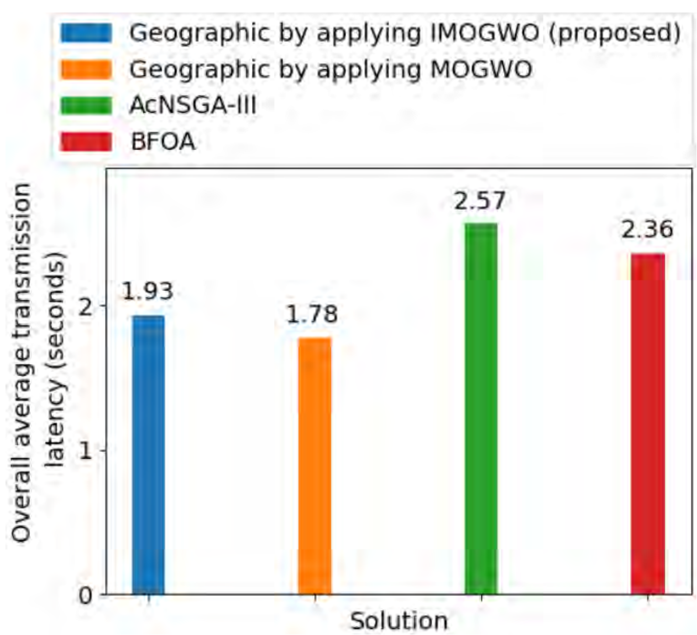


Figure 5.29: Comparison of the overall average transmission latency values for all the performed cycles



active objects also decreases over the cycles, reducing the number of available objects to make the various hops during the routing.

Figure 5.30: Comparison of the PDR values of over the performed cycles

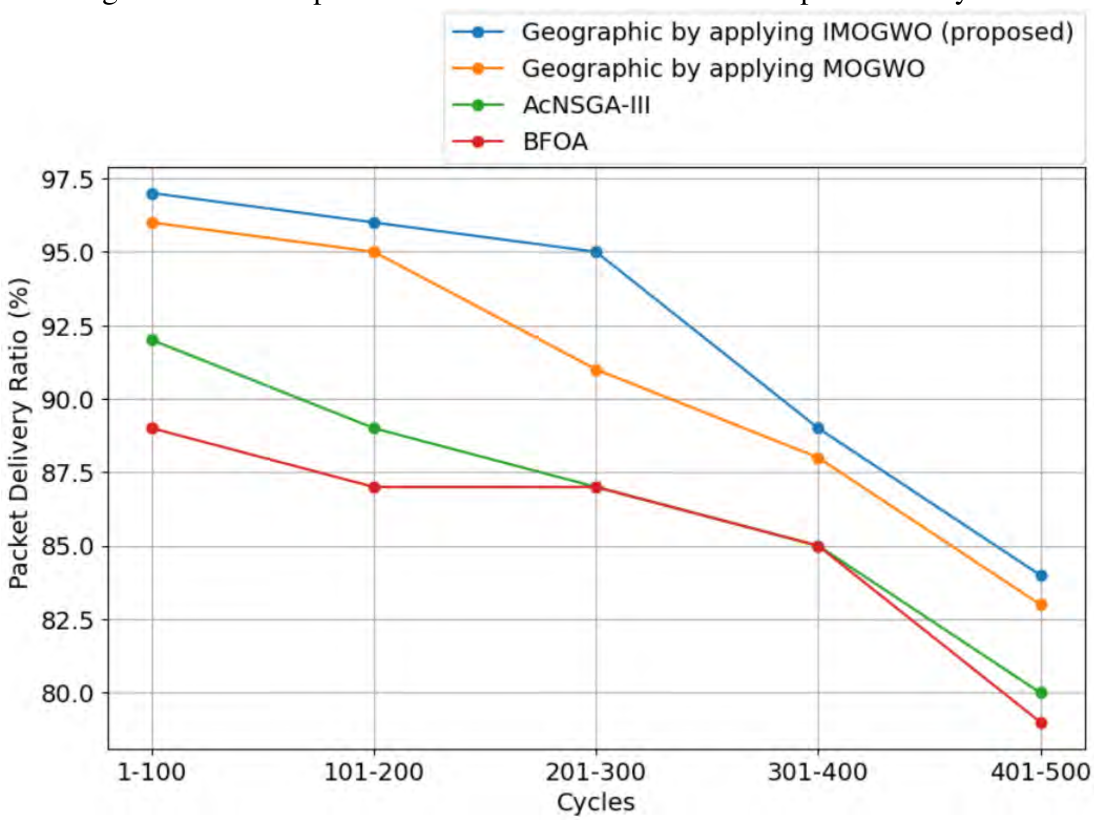


Figure 5.31: Comparison of the total PDR values calculated for all the cycles

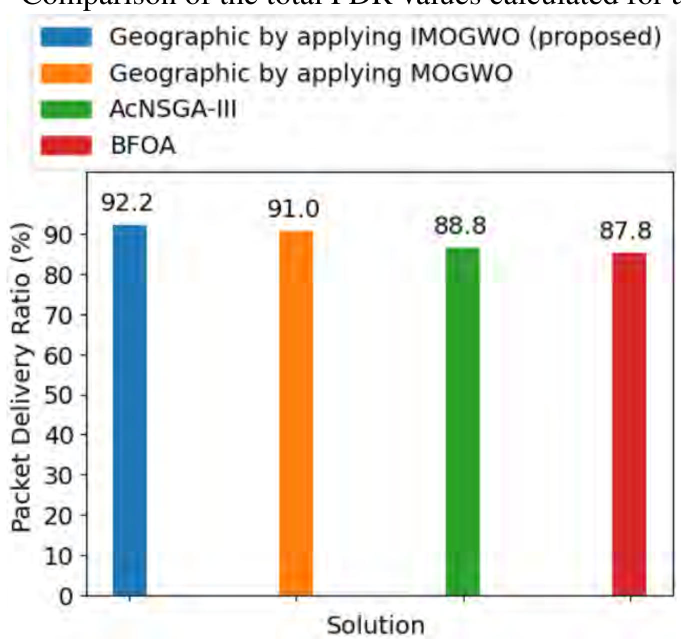


Figure 5.32 shows the variation in control Overhead across the cycles for the different evaluated routing algorithms. For our geographic routing approach with IMOGWO and MOGWO, as well as for the AcNSGA-III and BFOA algorithms, a general downward trend in control overhead across the cycles can be observed. This decrease may indicate an improved efficiency as the system gets adapted and learns about the objects in the network. In addition, this decrease can be justified by the decrease in the number of active objects in the network, particularly over the last few cycles. Furthermore, by analyzing the average control overhead values for all cycles shown in the Figure 5.33, it can be seen that the geographic routing approach with IMOGWO has a slightly higher overhead (21.98%) than that with MOGWO (21.25%). The AcNSGA-III and BFOA algorithms have control overheads of 20.52% and 18.59% respectively. This comparison shows that the BFOA algorithm is the most efficient in terms of control overhead minimization, closely followed by AcNSGA-III, where the Geographic routing approaches with the IMOGWO and the MOGWO show a slightly lower performance in terms of control overhead, but remain competitive with the other algorithms. This can be justified by the fact that our proposed geographic routing keeps more objects active for a higher number of cycles. In addition, it can be explained by the periodic exchange of 1-hop and 2-hop neighbor information applied in the proposed geographic routing. However, our proposal still performs slightly worse in terms of control Overhead than the other two algorithms, as it outperforms them in terms of packet transmission reliability (i.e. the amount of transmitted data is higher).

5.5.4.2 Comparison using inferential statistics tests

Indeed, to assess whether the proposed solution represents a significant improvement over other routing methods, we applied the inferential statistical tests presented in section 5.3.3. We applied tests involving (1 x N) comparisons. The Friedman test with Iman-Davenport's extension and Holm's post-hoc procedure are commonly used to evaluate the relative performance of a new algorithm compared to the existing solutions for a specific problem.

a) Application of the Friedman test with the Iman-Davenport extension

The Friedman test with the Iman-Davenport extension provides a method for comparing one algorithm with a set of other algorithms and statistically identifying significant differences between them. The steps involved in its application are detailed in section 5.3.3.1. Table 5.23 presents the statistical results of the Friedman test applied to compare our proposed geographic routing solution with the other routing algorithms studied with regard to the number of active objects and transmission latency. The lowest rank, indicating superior performance, is bolded, and the next rank is underlined.

For the criterion of the number of active objects in the network, we observe that our geographic routing proposal with IMOGWO obtains the lowest average rank among the four algorithms, with

Figure 5.32: Comparison of the control Overhead values

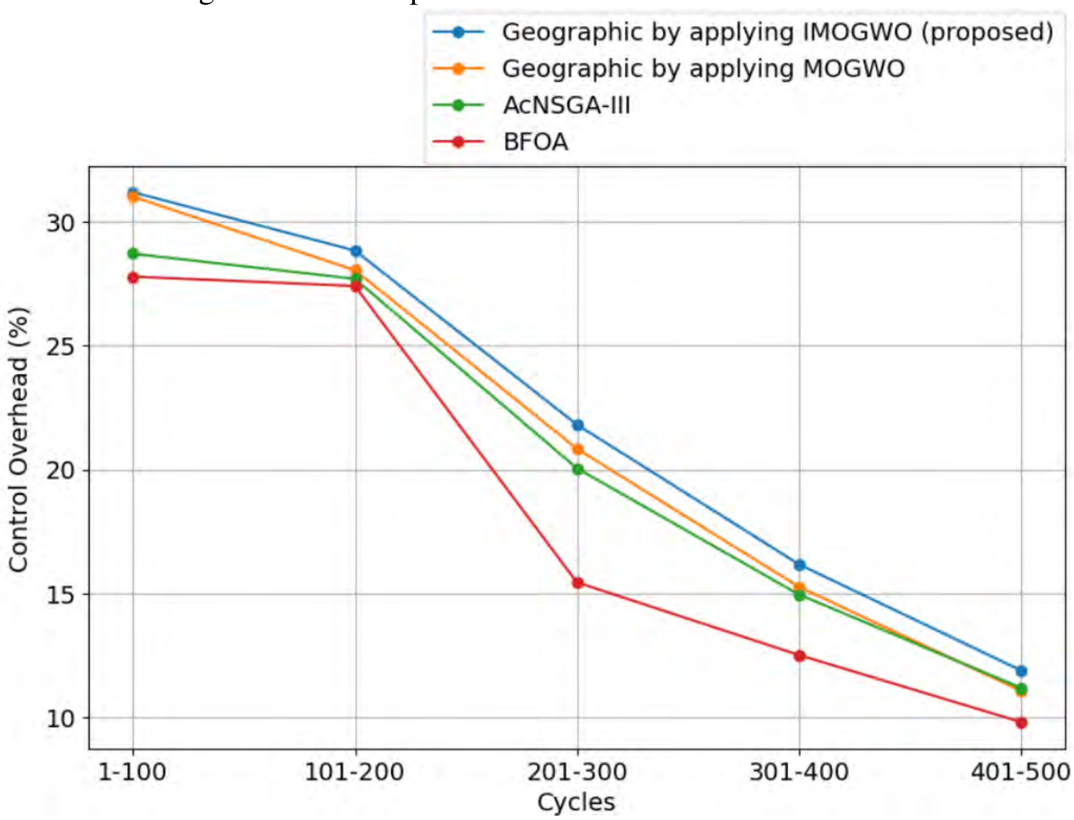
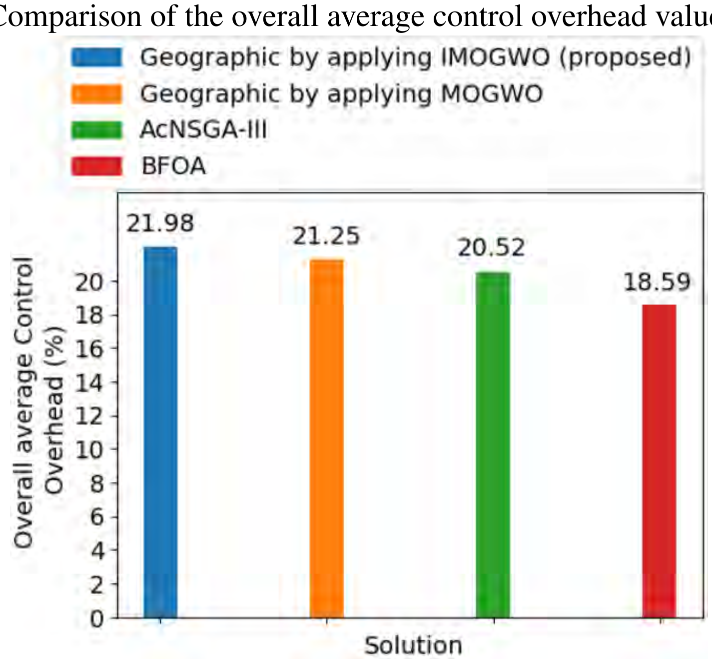


Figure 5.33: Comparison of the overall average control overhead values for all cycles



a rank of 1.42. This indicates that, on average, our approach with IMOGWO ranks higher than the other algorithms in terms of maintaining the number of active objects in the network. The proposed geographic routing algorithm applied with MOGWO has the second-highest ranking. On the other hand, the BFOA algorithm achieves the highest average rank, showing that it performs less well than the others in this context. Moreover, in terms of transmission latency, the geographic routing proposal with the IMOGWO obtains an average rank of 1.92, which is the second best result. However, the geographic routing proposal with the MOGWO shows the best result with an average rank of 1.13, while AcNSGA-III and BFOA algorithms perform worse, with average ranks of 3.8 and 3.07, respectively. In addition, the Friedman and Iman-Davenport statistics for the two criteria both indicate a statistically significant difference between the algorithms, with a value of F(3.499) = 2.653, confirming that performance varies significantly between the methods.
In fact, these analyses highlight the high performance of the geographic routing proposal with the IMOGWO, although slightly lower than that of the MOGWO in terms of transmission latency. The results also showed that the AcNSGA-III and BFOA algorithms present less satisfactory performances for the two evaluated criteria, which indicates the superiority of our geographical routing proposal based on multi-objective metaheuristics.

Table 5.23: Average ranks of routing algorithms and calculation of Friedman and Iman-Davenport statistics for metrics: number of active objects and transmission latency

Algorithm		Geographic with IMOGWO (proposed)	Geographic with MOGWO	AcNSGA-III	BFOA
Average Rank	Number of active objects	**1.42**	2.07	2.81	3.68
	Transmission latency	1.92	**1.13**	3.8	3.07
Friedman statistic for number of active objects = 861.41 Iman-Davenport statistic for number of active objects = 216.81 F(3,499) = 2.653					
Friedman statistic for transmission latency = 1321.02 Iman-Davenport statistic for transmission latency = 332.21 F(3,499) = 2.653					

b) Application of Holm's post-hoc procedure

The steps involved in applying this procedure are detailed in section 5.3.3.2. Table 5.24 represents the results of applying Holm's post-hoc procedure to the number of active objects in the network. It is remarkable that all the obtained z-values are negative, which confirms that the proposed solution outperforms all the other compared routing solutions. Moreover, our geographic routing proposal with IMOGWO obtains an average rank of 1, which places it in the lead, with a p-value of 0.0102. Then, the geographic routing proposal with the MOGWO ranks second with an average rank of 2 and a very significant p-value ($p < 0.0001$). The AcNSGA-III and BFOA algorithms obtain average ranks of 3 and 4, respectively, with highly significant p values ($p < 0.0001$ and $p = 1.0$).

Thus, these results confirm that the geographic routing proposal with the IMOGWO maintains a greater number of active objects in the network, indicating a better stability compared to the other evaluated methods.
With regard to the transmission latency criterion, Table 5.25 summarizes the results of Holm's post-hoc test for the comparison between the routing evaluated algorithms. These results also revealed significant differences between the algorithms. In this case, the geographic routing proposal with the MOGWO obtains the best average rank of 1, with a p-value of 0.0102. The geographic routing proposal with IMOGWO ranks second with an average rank of 2 and a highly significant p-value ($p < 0.0001$). The AcNSGA-III and BFOA algorithms ranked third and fourth respectively, with highly significant p-values ($p < 0.0001$ and $p = 1.0$). These results confirm that the geographic routing proposal with MOGWO has the lowest transmission latency of the methods evaluated, closely followed by the geographic routing proposal with IMOGWO.
In conclusion, the analysis of the results of the comparison of the routing algorithms highlights the differentiated performance of the different methods evaluated. Our geographic routing proposal with IMOGWO stands out by maintaining a higher number of active objects compared with the other methods, demonstrating better management of energy consumption and improved stability. As for transmission latency, our geographic routing approach developed with MOGWO has the lowest latency, closely followed by that applied with IMOGWO. These results highlight the effectiveness of our geographic and hybrid routing approach based on the IMOGWO and MOGWO optimization algorithms.

Table 5.24: Results of Holm's post-hoc test for comparison between routing algorithms in terms of the number of active objects

Algorithm	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
Geographic with IMOGWO	0	1	0.05	0.0102
Geographic with MOGWO	-8.009	2	0.025	< 0.0001
AcNSGA-III	-17.08	3	0.0167	< 0.0001
BFOA	-27.71	4	0.0125	1

Table 5.25: Results of Holm's post-hoc test for comparison between routing algorithms in terms of transmission latency

Algorithm	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
Geographic with IMOGWO	0	2	0.025	< 0.0001
Geographic with MOGWO	9.74	1	0.05	0.0102
AcNSGA-III	-23.72	4	0.0125	1.0
BFOA	-14.04	3	0.0167	< 0.0001

5.5.4.3 Synthesis

The obtained results highlight the superiority of our proposed geographic and hybrid routing approach over the other compared algorithms. This efficiency consists of optimized management of energy consumption, balanced load distribution between network nodes, and improved performance in terms of transmission latency and packet delivery rate. These observations can be justified by the adaptability of the geographic routing approach to dynamic changes in the network, as well as by the ability of the applied optimization algorithms, in particular IMOGWO, to find efficient transmission paths. Indeed, the results showed that IMOGWO outperforms MOGWO in solving the multi-objective geographic routing problem. In addition, the progressive decrease in the number of active objects in the network with the proposed geographic routing approach reflects better management of object energy, thus reducing the risk of energy depletion. Finally, although presenting a slight increase in terms of control data, the geographic routing approaches present a better reliability of packet transmission, thus ensuring a better quality of service throughout the network.

5.6 Conclusion

In conclusion, this chapter provides an overview of the numerical results, simulations and experiments carried out to evaluate the approaches proposed in this thesis. We have evaluated and compared our multi-objective reactive routing approach based on the MOGWO metaheuristic with NSGA-III. The results showed the superiority of our proposal over NSGA-III. Next, the effectiveness of our proposed algorithm, IMOGWO, in solving DTLZ benchmark problems was demonstrated through a series of tests. We also analyzed the performance of our indoor localization approach, which combines Deep Learning and meta-heuristics, and the results of the proposed hybrid multi-objective geographic routing algorithm. The results confirmed the superiority of our proposed solutions over the other solutions compared, and highlighted the strengths and weaknesses of the different evaluated approaches. These results will be summarized in the general conclusion (next chapter, section 6.1). The presented discussions and analyzes contribute to a better understanding of the performance of the proposed solutions and pave the way for future research in this exciting area.

Chapter 6

Conclusions and future research directions

Contents

In this thesis, we studied the challenges related to routing and indoor localization in IoT networks, which are two important process that ensure the efficiency and reliability of these constantly-evolving networks. These challenges were addressed using metaheuristic optimization and DL techniques. To achieve this, the thesis relies on an in-depth review of the literature, describing the context of routing and indoor localization in IoT networks.

The first contribution consists in a mathematical modeling of the routing problem in IoT networks. It provides a structured framework for the development and optimization of routing algorithms. Then, a multi-hop, reactive and multi-objective routing approach based on using the MOGWO algorithm [2] was introduced. Subsequently, an improved version of this algorithm, named IMOGWO, was proposed to solve problems having large number of objectives. A new method was also suggested to optimize the training of DL models using a metaheuristic. It is the fundamental element of a localization approach combining metaheuristic optimization with DL to accurately predict the positions of objects in an IoT network. Finally, a hybrid and multi-objective geographic routing approach was proposed. It leverages our approach of object localization to optimize the routing process by considering various objectives simultaneously.

The introduced approaches were evaluated by real experiments and simulations that demonstrated their best performance compared to other existing solutions. The proposed contributions and evaluation results are summarized in the following section.

6.1 Findings and results

The main contributions and the obtained findings are summarized below:

- We provided an in-depth literature review starting with a general introduction to IoT. Afterwards, we studied routing and indoor localization in IoT networks. Subsequently, we addressed fundamentals of optimization and DL. In the state of the art, recent work related to these domains were also presented.

- A mathematical modeling of routing in IoT networks was proposed. It defines and formalizes the characteristics of the IoT network, the key objectives, the decision variables and the constraints related to the routing problem. This modeling was applied to make the introduced approaches more structured and simplify their evaluation. It It focuses on various objectives: progress towards the final destination, efficient energy management, reliable connections between objects, route continuity and latency minimization.

- A routing approach, based on the MOGWO algorithm [2], was developed to treat routing as a multi-objective optimization problem. Our main objective was to design a multi-hop, reactive and multi-objective routing algorithm in IoT network having hybrid topology. The performance of this routing approach was evaluated, through real experiments, and compared to that of NSGA-III [59]. The obtained results highlight that MOGWO offers a more efficient and promising solution for routing in IoT networks than NSGA-III. This approach guarantees better energy efficiency, improved load balancing, reduced transmission delays, more reliable and stable communications and more efficient distribution of objects in the network.

- An enhanced version of the MOGWO multi-objective metaheuristic, called Improved MOGWO (IMOGWO) was also suggested. It adjusts the fundamental equations applied by MOGWO. It aims at increasing the efficiency of MOGWO in solving complex problems with multiple objectives, ensuring efficient convergence and optimal distribution of the obtained solutions. Our objective is to develop a multi-objective geographic routing in IoT networks based on IMOGWO. As a preliminary evaluation, its performance was tested by solving the DTLZ [4] benchmark problems. A series of tests were performed on these problems by varying the number of objectives and the studied problem. The findings provided by IMOGWO were compared with those obtained by the following optimizers (MOGWO, DMOPSO [63], MOEA/D-IEps [69], MOEA/DD [66] and ESPEA [64]), applying IGD and NHV indicators. Standard comparisons and inferential statistical tests were carried out to show that IMOGWO presents more efficient convergence and optimal distribution of the obtained solutions, in comparison with other optimizers.

- A novel technique was introduced to optimize the training of DL models utilizing a single-objective metaheuristic. Its originality lies in the fact that the developed optimizer can be applied to several DL techniques, which makes it adaptable to different neural network architectures. The method was, then, utilized to develop the indoor localization approach in IoT networks.

- An indoor localization approach, based on our previously-suggested approach, employed in IoT networks, was proposed. It trains a DL model with UWB ToF measurements by applying the GWO metaheuristic optimization algorithm [3]. The performance of our solution was compared to that of three other widely-used localization techniques: a method based on DL [154], the IWCL [155] and a multilateration technique [156]. To demonstrate the superiority of the introduced approach, standard comparisons and inferential statistical tests were conducted. The comparison results revealed that the developed localization approach ensures better and more accurate estimation of the positions and outperforms other tested localization solutions. Indeed, the suggested method provided an average error of only 0.057 meters between the actual and predicted positions and a localization accuracy of 98.92%. These results are better than those provided by other recent indoor localization solutions which gave an average error of tens of centimeters.

- We have developed a hybrid, multi-objective geographic routing approach designed for IoT networks having mesh architecture. The geographic nature of the proposed algorithm requires determining the positions of objects in the network, which is provided by our proposed indoor localization approach. The multi-hop route selection process was optimized according to various objectives simultaneously using the proposed IMOGWO algorithm. Hybrid simulations, combining simulated and real objects, were performed to evaluate the proposed geographic and hybrid routing approach. The performance of our routing approach relying on IMOGWO was compared to that of the same approach based on MOGWO, a QoS-based routing algorithm using AcNSGA-III [83] and a geographic routing algorithm relying on BFO [109]. This comparison was performed employing various evaluation criteria such as efficiency, reliability, energy optimization, load balancing and stability. Standard comparisons and analyses by inferential statistical tests showed that IMOGWO outperforms MOGWO in solving the multi-objective geographic routing problem. They also confirmed the superiority of our geographic and hybrid routing approach over other compared algorithms. The proposed approach exhibited high its effectiveness in energy consumption management, load balancing, transmission latency and packet delivery rate. Despite a slight increase in control data, geographic routing approaches, using MOGWO and IMOGWO, provide better transmission reliability and ensure, consequently, better quality of service throughout the network.

To sum up, the evaluations of the suggested approaches through real experiments and hybrid simulations, the use of the adapted performance indicators, its comparisons with the existing solutions and the conducted statistical analyzes highlight the high effectiveness of our approaches in routing and indoor localization.

6.2 Future research directions

In order to explore in depth the future research directions and practical implications of this thesis, we discuss, in this section, its future perspectives.

- The deployment of the proposed localization and routing approaches and their implementation in the LocURa4IoT platform [140] by considering hardware constraints and performance requirements.

- The extension of the theoretical aspects: extending the mathematical and theoretical modeling of routing and localization problems in an indoor environment by exploring new theoretical frameworks.

- Optimization of the IMOGWO algorithm: to reduce the execution time and the complexity of the calculations in the introduced IMOGWO metaheuristic, the processes of updating the positions of the wolves will be to modified. For example, we will consider in-depth analysis of the searching and exploration phases of the algorithm to better optimize the selection of leader wolves and enhance the management of non-dominated solutions. By making these adjustments, we intend to develop an improved version of IMOGWO that allows reducing the execution time and the computational complexity and, consequently, provides more competitive solution to deal with multi-objective problems.

- Self-learning and self-adapting of DL model: designing a localization system that can detect and react to changes in the environment resulting from adding or moving anchors and the presence of temporary obstacles. The objective is to develop a DL model capable of self-learning and self-adapting to changes in the environment and network conditions. In other words, this model does not need to be retrained completely when there are changes in the environment. However, it would continuously evolve as it is exposed to new data, ensuring real-time localization and dynamic adaptation to new scenarios. This adaptability will allow the system to maintain precise and reliable localization despite the modification in the network topology.

- Multi-modal data fusion: Investigate methods to efficiently fuse data provided by different sensory sources (e.g. UWB, Wi-Fi, Bluetooth, etc.) using DL techniques in order to improve indoor location accuracy in IoT networks.

- Artificial Intelligence-based routing: integrate Artificial Intelligence techniques, such as reinforcement learning, into the routing algorithms to improve the efficiency and adaptability of routing in IoT networks. Reinforcement learning is a sub-field of machine learning where an agent learns to make decisions by being interacted into an environment. The agent's objective is to maximize a long-term cumulative reward by choosing appropriate actions made in different situations.

- Big Data analysis to optimized routing and indoor location: study the use of Big Data techniques to process real-time stream data, such as Apache Kafka [160] or Apache Flink [161], to continuously analyze, in real-time, data generated by IoT networks. This data may include information about location, routing, traffic and quality of service.

- Edge Computing: explore the opportunities offered by Edge Computing to improve the performance of routing and indoor localization in IoT networks. Edge Computing involves data processing and decision-making nearby sensors. Unlike the traditional models, where data is sent to a central point or to the cloud to be processed, Edge Computing moves some of that processing to the ends of the network, which reduces the latency and dependence on centralization and cloud.

- Specific applications: focus on the specific applications of routing and indoor localization solutions in areas, such as healthcare, logistics, smart cities and smart buildings, by identifying relevant use cases and adapting the algorithms by considering the characteristics and constraints specific to each context.

- Multi-objective optimization: conduct further research on multi-objective optimization techniques to address routing and indoor localization challenges in IoT networks.

Publications (within this thesis)

International Journals

- S. TLILI, S. MNASRI and T. VAL. **The Internet of Things enabling communication technologies, applications and challenges: a survey**. International Journal of Wireless and Mobile Computing. ISSN 1741-1084, 2022, vol. 23, no 1, p. 9-21.
- S. TLILI, S. MNASRI and T. VAL. **An improved multi-objective Grey Wolf Optimizer**. Natural Computing. (https://link.springer.com/journal/11047) ISSN 1567-7818 (Print) 1572-9796 (Electronic). (Impact Factor= 2.1), [Submission under peer review, submitted on November 12, 2022].
- S. TLILI, S. MNASRI and T. VAL. **UWB Time of Flight-based indoor IoT localisation solution with Deep Learning optimised by metaheuristics**. International Journal of Sensor Networks. ISSN 1748-1279, 2024, vol. 44, no 2, p. 99-123.
- S. TLILI, S. MNASRI and T. VAL. **Edge-computing-enabled hybrid and multi-objective geographic routing for mesh IoT networks: an IMOGWO-based approach**. Simulation Modelling Practice and Theory. (https://www.sciencedirect.com/journal/simulation-modelling-practice-and-theory) ISSN 1569-190X (Print) 1878-1462 (Online). (Impact Factor= 3.5) [Submission under peer review, submitted on June 27, 2024]
- S. TLILI, S. MNASRI and T. VAL. **Arithmetic Meta-Heuristic Optimization for Enhanced Deep Learning: A TCN-Based Language Modeling Approach**. Computational Science and Engineering. (https://www.inderscience.com/jhome.php?jcode=ijcse) ISSN 1742-7185 (Print) 1742-7193 (Online). (Impact Factor= 1.4) [Submission under peer review, submitted on August 27, 2024]

International Conferences

- S. TLILI, S. MNASRI and T. VAL. **A survey on IoT Routing: Types, Challenges and Contribution of Recent Used Intelligent Methods**. 2022 2nd International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, 2022, pp. 161-166, doi: 10.1109/ICCIT52419.2022.9711649.

National Conferences

- S. TLILI, S. MNASRI and T. VAL. **A multi-objective Gray Wolf algorithm for routing in IoT Collection Networks with real experiments**. 2021 National Computing Colleges Conference (NCCC), Taif, Saudi Arabia, 2021, pp. 1-5, doi: 10.1109/NCCC49330.2021.9428865.

Bibliography

[1] Number of iot devices 2015-2025 | statista. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. Accessed: 2021-05-17.

[2] Seyedali Mirjalili, Shahrzad Saremi, Seyed Mohammad Mirjalili, and Leandro dos S. Coelho. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems with Applications*, 47:106 – 119, 2016.

[3] Seyedali Mirjalili, Seyed Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in Engineering Software*, 69:46 – 61, 03 2014.

[4] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. *Mechanical Engineering*, 01 2008.

[5] Cisco. Cisco visual networking index: Forecast and trends, 2017-2022. *Cisco public White Paper*, February 2019.

[6] Build with matter | smart home device solution. https://csa-iot.org/all-solutions/matter/, 2024. Accessed: 2024-04-24.

[7] Sihem Tlili, Mnasri Sami, and Thierry Val. The internet of things enabling communication technologies, applications and challenges: a survey. *International Journal of Wireless and Mobile Computing*, 23:9–21, 03 2022.

[8] Sihem Tlili, Sami Mnasri, and Thierry Val. A survey on iot routing: Types, challenges and contribution of recent used intelligent methods. In *2022 2nd International Conference on Computing and Information Technology (ICCIT)*, pages 161–166, 2022.

[9] Mamtha M. Pandith, Nataraj Kanathur Ramaswamy, Mallikarjunaswamy Srikantaswamy, and Rekha Kanathur Ramaswamy. A comprehensive review of geographic routing protocols in wireless sensor network, 2022.

[10] Allam Balaram, Manda Silparaj, Shaik Nabi, and P. Chandana. *A Comprehensive Survey of Geographical Routing in Multi-hop Wireless Networks*, pages 141–172. 01 2022.

[11] A. Roshini and K.V.D. Kiran. Hierarchical energy efficient secure routing protocol for optimal route selection in wireless body area networks. *International Journal of Intelligent Networks*, 4:19–28, 2023.

[12] Md Aminul Islam, Muhammad Ismail, Rachad Atat, Osman Boyaci, and Susmit Shannigrahi. Software-defined network-based proactive routing strategy in smart power grids using graph neural network and reinforcement learning. *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, 5:100187, 2023.

[13] Stephen Mak, Liming Xu, Tim Pearce, Michael Ostroumov, and Alexandra Brintrup. Fair collaborative vehicle routing: A deep multi-agent reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, 157:104376, 2023.

[14] Armando Gomes. Coalitional bargaining games: A new concept of value and coalition formation. *Games and Economic Behavior*, 132:463–477, 2022.

[15] Mingyue Zhang, Jianpeng Xie, Zongyang Wang, Lutong Liang, Pengfei Gu, Peilin Jin, and Jie Zhou. Mo-cbacorp: A new energy-efficient secure routing protocol for underwater monitoring wireless sensor network. *Journal of King Saud University - Computer and Information Sciences*, 35(9):101786, 2023.

[16] Atonu Ghosh, Sudip Misra, Venkanna Udutalapally, and Debanjan Das. Loraute: Routing messages in backhaul lora networks for underserved regions. *IEEE Internet of Things Journal*, 10(22):19964–19971, 2023.

[17] Zhiqun Wang, Zikai Jin, Zhen Yang, Wenchao Zhao, and Mohammad Trik. Increasing efficiency for routing in internet of things using binary gray wolf optimization and fuzzy logic. *Journal of King Saud University - Computer and Information Sciences*, 35(9):101732, 2023.

[18] Amine Kardi and Rachid Zagrouba. Esharp: Energy-efficient and smart hierarchical routing protocol based on smart slumber for wireless sensor networks. *Wireless Personal Communications*, 123:1809–1824, 03 2022.

[19] Rachid Zagrouba and Amine Kardi. Hierarchical smart routing protocol for wireless sensor networks. *International Journal of Computer Applications in Technology*, 65:78, 01 2021.

[20] Prabhakar Reddy B, Bhaskar Reddy B, and Dhananjaya B. The aodv routing protocol with built-in security to counter blackhole attack in manet. *Materials Today: Proceedings*, 50:1152–1158, 2022. 2nd International Conference on Functional Material, Manufacturing and Performances (ICFMMP-2021).

[21] Charles Perkins and Elizabeth Belding. Ad-hoc on-demand distance vector routing. volume 25, pages 90–100, 01 1999.

[22] M. Venkatanaresh, Ranjeet Yadav, D. Thiyagarajan, S. Yasotha, Gowtham Ramkumar, and Penumathsa Suresh Varma. Effective proactive routing protocol using smart nodes system. *Measurement: Sensors*, 24:100456, 2022.

[23] Wael Ali Hussein, Borhanuddin M Ali, MFA Rasid, and Fazirulhisyam Hashim. Smart geographical routing protocol achieving high qos and energy efficiency based for wireless multimedia sensor networks. *Egyptian Informatics Journal*, 23(2):225–238, 2022.

[24] Ohoud Alzamzami and Imad Mahgoub. Geographic routing enhancement for urban vanets using link dynamic behavior: A cross layer approach. *Vehicular Communications*, 31:100354, 2021.

[25] Weifeng Sun, Zun Wang, and Guanghao Zhang. A qos-guaranteed intelligent routing mechanism in software-defined networks. *Computer Networks*, 185:107709, 2021.

[26] Ahmad Raza Hameed, Saif ul Islam, Mohsin Raza, and Hasan Ali Khattak. Towards energy and performance-aware geographic routing for iot-enabled sensor networks. *Computers and Electrical Engineering*, 85:106643, 2020.

[27] Alwi M. Bamhdi. Efficient dynamic-power aodv routing protocol based on node density. *Computer Standards and Interfaces*, 70:103406, 2020.

[28] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low-cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34, 2000.

[29] D. Niculescu and B. Nath. Ad hoc positioning system (aps). In *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, volume 5, pages 2926–2931 vol.5, 2001.

[30] Qing Zhao, Zhen Xu, and Lei Yang. An improvement of dv-hop localization algorithm based on cyclotomic method in wireless sensor networks. *Applied Sciences*, 13:3597, 03 2023.

[31] Huanqing Cui, Sen Wang, and Chuanai Zhou. A high-accuracy and low-energy range-free localization algorithm for wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2023, 05 2023.

[32] Abdelali Hadir, Naima Kaabouch, Mohammed-Alamine El Houssaini, and Jamal El Kafi. Range-free localization approaches based on intelligent swarm optimization for internet of things. *Information*, 14(11), 2023.

[33] Tian He, Chengdu Huang, Brain Blum, John Stankovic, and Tarek Abdelzaher. Range-free localization schemes for large scale sensor networks. *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, pages 81–95, 04 2003.

[34] L. Doherty, K.S.J. pister, and L. El Ghaoui. Convex position estimation in wireless sensor networks. In *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, volume 3, pages 1655–1663 vol.3, 2001.

[35] Trifun Savic, Xavier Vilajosana, and Thomas Watteyne. Constrained localization: A survey. *IEEE Access*, 10:1–1, 05 2022.

[36] Satish Jondhale, Maheswar Rajagopal, and Jaime Lloret. *Survey of Existing RSSI-Based LT Systems*, pages 49–64. 01 2022.

[37] Mohammed A. A. Al-qaness, Mohamed Elsayed Abd Elaziz, Ahmed Ewees, Aaqif Abbasi, Yousif Alhaj, and Ammar Hawbani. Channel state information from pure communication to sense and track human motion: A survey. *Sensors*, 19:3329, 07 2019.

[38] Francois Despaux, Adrien van den Bossche, Katia Jaffres-Runser, and Thierry Val. N-twr: An accurate time-of-flight-based n-ary ranging protocol for ultra-wide band. *Ad Hoc Networks*, 79, 05 2018.

[39] Ashish Garg and Amit Gupta. Indoor tracking using ble -brief survey of techniques. 03 2020.

[40] Toan Dinh, Thanh Nguyen, Hoang-Phuong Phan, Van Dau, Dzung Dao, and Nam-Trung Nguyen. Physical sensors: Thermal sensors. In Roger Narayan, editor, *Encyclopedia of Sensors and Biosensors (First Edition)*, pages 20–33. Elsevier, Oxford, first edition edition, 2023.

[41] Thomas Hillebrandt, Heiko Will, and Marcel Kyas. The membership degree min-max localisation algorithm, 2023.

[42] Xu Feng, Khuong An Nguyen, and Zhiyuan Luo. A survey of deep learning approaches for wifi-based indoor positioning. *Journal of Information and Telecommunication*, 6(2):163–216, 2022.

[43] Toni Perkovic, Lea Dujic Rodic, Josip Sabic, and Petar Solic. Machine learning approach towards lorawan indoor localization. *Electronics*, 12:457, 01 2023.

[44] Xiaoyan Shen, Boyang Xu, and Hongming Shen. Indoor localization system based on rssi-apit algorithm. *Sensors*, 23:9620, 12 2023.

[45] Satish Jondhale, Vijay Mohan, Bharat Sharma, Jaime Lloret, and Shashikant Athawale. Support vector regression for mobile target localization in indoor environments. *Sensors*, 22:358, 01 2022.

[46] Liping Wang and Sudeep Pasricha. A framework for csi-based indoor localization with id convolutional neural networks. In *2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8. IEEE, 2022.

[47] Mao Li, Feng Jiang, and Cong Pei. Improvement of triangle centroid localization algorithm based on pit criterion (itcl-pit) for wsns. *EURASIP Journal on Wireless Communications and Networking*, 2022, 03 2022.

[48] Tinh Pham and Ta Mai. Ensemble learning model for wifi indoor positioning systems. *IAES International Journal of Artificial Intelligence (IJ-AI)*, 10:200, 03 2021.

[49] Danshi Sun, Erhu Wei, Zhuoxi Ma, Chenxi Wu, and Shiyi Xu. Optimized cnns to indoor localization through ble sensors using improved pso. *Sensors*, 21(6), 2021.

[50] Quentin Vey, Francois Spies, Baptiste Pestourie, Denis Genon-Catalot, Adrien van den Bossche, Rejane Dalce, JULIEN SCHRIVE, and Thierry Val. Poucet: A multi-technology indoor positioning solution for firefighters and soldiers. 11 2021.

[51] Beiya Yang, Erfu Yang, Leijian Yu, and Andrew Loeliger. High-precision uwb-based localisation for uav in extremely confined environments. *IEEE Sensors Journal*, 22(1):1020–1029, 2022.

[52] C. Lamoureux and R. Chelouah. Fusion particle and fingerprint recognition for indoor positioning system on mobile. *Engineering Applications of Artificial Intelligence*, 98:104082, 2021.

[53] Guannan Liu, Hsiao-Chun Wu, Weidong Xiang, Jinwei Ye, Yiyan Wu, and Limeng Pu. Indoor object localization and tracking using deep learning over received signal strength. In *2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pages 1–6, 2020.

[54] Quentin Vey, Rejane Dalce, Adrien van den Bossche, and Thierry Val. A distributed algorithm for range-based localization in sparse wireless networks. In *2020 30th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–8, 2020.

[55] Brahim El Boudani, Loizos Kanaris, Akis Kokkinis, Michalis Kyriacou, Christos Chrysoulas, Stavros Stavrou, and Tasos Dagiuklas. Implementing deep learning techniques in 5g iot networks for 3d indoor positioning: Delta (deep learning-based co-operative architecture). *Sensors*, 20, 09 2020.

[56] Paschal Nyiam and Abdel Salhi. On the simplex, interior-point and objective space approaches to multiobjective linear programming. *Journal of Algorithms and Computational Technology*, 15:174830262110084, 10 2021.

[57] Jean-Francois Cordeau, Fabio Furini, and Ivana Ljubic. Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research*, 275(3):882–896, 2019.

[58] Mohammad H. Nadimi-Shahraki, Shokooh Taghian, and Seyedali Mirjalili. An improved grey wolf optimizer for solving engineering problems. *Expert Systems with Applications*, 166, 02 2021.

[59] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *Evolutionary Computation, IEEE Transactions on*, 18:602–622, 08 2014.

[60] Ahmed Gad. Particle swarm optimization algorithm and its applications: A systematic review. *Archives of Computational Methods in Engineering*, 29:2531–2561, 04 2022.

[61] Feng Pan, Xiaohui Hu, Russ Eberhart, and Yaobin Chen. An analysis of bare bones particle swarm. pages 1 – 5, 10 2008.

[62] Yudong Zhang, Shuihua Wang, and Genlin ji. A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical Problems in Engineering*, 2015:1–38, 10 2015.

[63] Ki-Baek Lee and Jong-Hwan Kim. Dmopso: Dual multi-objective particle swarm optimization. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 3096–3102, 2014.

[64] Marlon Braun, Pradyumn Kumar Shukla, and Hartmut Schmeck. Obtaining optimal pareto front approximations using scalarized preference information. pages 631–638, July 2015. Copyright: Copyright 2017 Elsevier B.V., All rights reserved.; 16th Genetic and Evolutionary Computation Conference, GECCO 2015 ; Conference date: 11-07-2015 Through 15-07-2015.

[65] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[66] Qian Xu, Zhanqi Xu, and Tao Ma. A survey of multiobjective evolutionary algorithms based on decomposition: Variants, challenges and future directions. *IEEE Access*, 8:41588–41614, 2020.

[67] Ram Agrawal, Kalyanmoy Deb, and Ram Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:1–34, 06 2000.

[68] Kalyanmoy Deb, Mayank Goyal, et al. A combined genetic adaptive search (geneas) for engineering design. *Computer Science and informatics*, 26:30–45, 1996.

[69] Zhun Fan, Wenji Li, Xinye Cai, Han Huang, Yi Fang, You Yugen, Jiajie Mo, Caimin Wei, and Erik Goodman. An improved epsilon constraint-handling method in moea/d for cmops with large infeasible regions. *Soft Computing*, 23, 12 2019.

[70] Zhixiang Yang, Xinye Cai, and Zhun Fan. Epsilon constrained method for constrained multiobjective optimization problems: Some preliminary results. *GECCO 2014 - Companion Publication of the 2014 Genetic and Evolutionary Computation Conference*, 07 2014.

[71] Muhammad Asif Jan and Rashida Adeeb Khanum. A study of two penalty-parameterless constraint handling techniques in the framework of moea/d. *Applied Soft Computing*, 13(1):128–148, 2013.

[72] Bernardo Morales-Castañeda, Daniel Zaldívar, Erik Cuevas, Oscar Maciel-Castillo, Itzel Aranguren, and Fernando Fausto. An improved simulated annealing algorithm based on ancient metallurgy techniques. *Applied Soft Computing*, 84:105761, 2019.

[73] Hesamoddin Tahami and Hengameh Fakhravar. A literature review on combining heuristics and exact algorithms in combinatorial optimization. *European Journal of Information Technologies and Computer Science*, 2:6–12, 04 2022.

[74] Moch Umam, Mustafid Mustafid, and Suryono Suryono. A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. *Journal of King Saud University - Computer and Information Sciences*, 34, 09 2021.

[75] H Sabireen and Neelanarayanan Venkataraman. A hybrid and light weight metaheuristic approach with clustering for multi-objective resource scheduling and application placement in fog environment. *Expert Systems with Applications*, 223:119895, 2023.

[76] Charles Audet, Jean Bigeon, Dominique Cartier, Sebastien Le Digabel, and Ludovic Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 292(2):397–422, 2021.

[77] R. While, Lucas Bradstreet, and Luigi Barone. A fast way of calculating exact hypervolumes. *IEEE Trans. Evolutionary Computation*, 16:86–95, 02 2012.

[78] Riccardo Albertin, Alessandro Prada, and Andrea Gasparella. A novel efficient multi-objective optimization algorithm for expensive building simulation models. *Energy and Buildings*, 297:113433, 2023.

[79] To Thanh Binh and Ulrich Korn. Mobes: A multiobjective evolution strategy for constrained optimization problems. In *The third international conference on genetic algorithms (Mendel 97)*, volume 25, page 27, 1997.

[80] N Palli, P McCluskey, S Azarm, and R Sundararajan. An interactive multistage ε-inequality constraint method for multiple objectives decision making. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 80326, page V002T02A006. American Society of Mechanical Engineers, 1998.

[81] B. K. Kannan and S. N. Kramer. An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design. *Journal of Mechanical Design*, 116(2):405–411, 06 1994.

[82] Saeed Fallahi and Mohamadreza Taghadosi. Quantum-behaved particle swarm optimization based on solitons. *Scientific Reports*, 12:13977, 08 2022.

[83] Sami Mnasri and Malek Alrashidi. Energy-efficient iot routing based on a new optimizer. *Simulation Modelling Practice and Theory*, 119:102591, 2022.

[84] Akshaya Kumar Mandal and Satchidananda Dehuri. *A Survey on Ant Colony Optimization for Solving Some of the Selected NP-Hard Problem*, pages 85–100. 01 2020.

[85] Laith Abualigah, Ali Diabat, Seyedali Mirjalili, Mohamed Abd Elaziz, and Amir H. Gandomi. The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, 376:113609, 2021.

[86] Juan Zou, Zhenghui Zhang, Jinhua Zheng, and Shengxiang Yang. A many-objective evolutionary algorithm based on dominance and decomposition with reference point adaptation. *Knowledge-Based Systems*, 231:107392, 2021.

[87] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on evolutionary computation*, 18(4):602–622, 2013.

[88] Ran Cheng, Miqing Li, Ye Tian, Xingyi Zhang, Yaochu Jin, and Xin Yao. A benchmark test suite for evolutionary many-objective optimization. *Complex and Intelligent Systems*, 3:67–81, 03 2017.

[89] Miqing Li, Crina Grosan, Shengxiang Yang, Xiaohui Liu, and Xin Yao. Multiline distance minimization: A visualized many-objective test problem suite. *IEEE Transactions on Evolutionary Computation*, 22(1):61–78, 2017.

[90] Mario Köppen and Kaori Yoshida. Substitute distance assignments in nsga-ii for handling many-objective optimization problems. pages 727–741, 01 2006.

[91] J. Anila Sharon, A. Hepzibah Christinal, D. Abraham Chandy, and Chandrajit Bajaj. Chapter 11 - application of intelligent edge computing and machine learning algorithms in mbti personality prediction. In D. Jude Hemanth, Brij B. Gupta, Mohamed Elhoseny, and Swati Vijay Shinde, editors, *Intelligent Edge Computing for Cyber Physical Applications*, Intelligent Data-Centric Systems, pages 187–215. Academic Press, 2023.

[92] Xin Wang, Liting Yan, and Qizhi Zhang. Research on the application of gradient descent algorithm in machine learning. In *2021 International Conference on Computer Network, Electronic and Automation (ICCNEA)*, pages 11–15, 2021.

[93] Xin Gao, Fang Deng, Hao Zheng, Ning Ding, Ziman Ye, Yeyun Cai, and Xiangyang Wang. Followed the regularized leader (ftrl) prediction model based photovoltaic array reconfiguration for mitigation of mismatch losses in partial shading condition. *IET Renewable Power Generation*, 16, 09 2021.

[94] Madhusmita Sahu and Rasmita Dash. *A Survey on Deep Learning: Convolution Neural Network (CNN)*, pages 317–325. 01 2021.

[95] Benedikt Bollig, Martin Leucker, and Daniel Neider. *A Survey of Model Learning Techniques for Recurrent Neural Networks*, pages 81–97. Springer Nature Switzerland, Cham, 2022.

[96] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014.

[97] D. Elhani, A.C. Megherbi, A. Zitouni, F. Dornaika, S. Sbaa, and A. Taleb-Ahmed. Optimizing convolutional neural networks architecture using a modified particle swarm optimization for image classification. *Expert Systems with Applications*, 229:120411, 2023.

[98] Burak Gülmez. Stock price prediction with optimized deep lstm network with artificial rabbits optimization algorithm. *Expert Systems with Applications*, 227:120346, 2023.

[99] Liying Wang, Qingjiao Cao, Zhenxing Zhang, Seyedali Mirjalili, and Weiguo Zhao. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. *Engineering Applications of Artificial Intelligence*, 114:105082, 2022.

[100] Dixizi Liu, Weiping Ding, Zhijie Sasha Dong, and Witold Pedrycz. Optimizing deep neural networks to predict the effect of social distancing on covid-19 spread. *Computers and Industrial Engineering*, 166:107970, 2022.

[101] Abdelaziz Abdelhamid, El-Sayed El-kenawy, Bandar Alotaibi, Ghada Amer, Mahmoud Abdelkader, Abdelhameed Ibrahim, and Marwa Eid. Robust speech emotion recognition using cnn+lstm based on stochastic fractal search optimization algorithm. *IEEE Access*, 10:1–21, 05 2022.

[102] Abdelhameed Ibrahim, Seyedali Mirjalili, M. El-Said, Sherif S. M. Ghoneim, Mosleh M. Al-Harthi, Tarek F. Ibrahim, and El-Sayed M. El-Kenawy. Wind speed ensemble forecasting based on deep learning using adaptive dynamic optimization algorithm. *IEEE Access*, 9:125787–125804, 2021.

[103] Rasmiranjan Mohakud and Rajashree Dash. Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection. *Journal of King Saud University - Computer and Information Sciences*, 34(8, Part B):6280–6291, 2022.

[104] Hamza Turabieh. A Hybrid Convolution Neural Network with Binary Particle Swarm Optimization for Intrusion Detection. *International Journal of Computer Science and Information Security (IJCSIS)*, 18, January 2021.

[105] Swarna Priya R.M., Praveen Kumar Reddy Maddikunta, Parimala M., Srinivas Koppu, Thippa Reddy Gadekallu, Chiranji Lal Chowdhary, and Mamoun Alazab. An effective feature engineering for dnn using hybrid pca-gwo for intrusion detection in iomt architecture. *Computer Communications*, 160:139–149, 2020.

[106] X.J. Luo, Lukumon O. Oyedele, Anuoluwapo O. Ajayi, Olugbenga O. Akinade, Hakeem A. Owolabi, and Ashraf Ahmed. Feature extraction and genetic algorithm enhanced adaptive deep neural network for energy consumption prediction in buildings. *Renewable and Sustainable Energy Reviews*, 131:109980, 2020.

[107] Sadeq D. Al-Majidi, Maysam F. Abbod, and Hamed S. Al-Raweshidy. A particle swarm optimisation-trained feedforward neural network for predicting the maximum power point of a photovoltaic array. *Engineering Applications of Artificial Intelligence*, 92:103688, 2020.

[108] Yulong Wang, Zhang Haoxin, and Guangwei Zhang. cpso-cnn: An efficient pso-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. *Swarm and Evolutionary Computation*, 49:114–123, 09 2019.

[109] Shreyas J, Chethana Reddy, Dharamendra Chouhan, P. K. Udayaprasad, N. N. Srinidhi, and S. M. Dilipkumar. Geographic routing scheme for resource and communication efficiency in the iot ecosystem using swarm-intelligence based bfo algorithm. *Journal of Information Technology Management*, 14(1):41–64, 2022.

[110] Chen Guo, Heng Tang, Ben Niu, and Chang Boon Patrick Lee. A survey of bacterial foraging optimization. *Neurocomputing*, 452:728–746, 2021.

[111] Archana Ojha and Prasenjit Chanak. Multiobjective gray-wolf-optimization-based data routing scheme for wireless sensor networks. *IEEE Internet of Things Journal*, 9(6):4615–4623, 2022.

[112] Mohamed Elshrkawey, Hassan Al-Mahdi, and Walid Atwa. An enhanced routing algorithm based on a re-position particle swarm optimization (ra-rpso) for wireless sensor network. *Journal of King Saud University - Computer and Information Sciences*, 34(10, Part B):10304–10318, 2022.

[113] Reena Pingale and S. Shinde. Multi-objective sunflower based grey wolf optimization algorithm for multipath routing in iot network. *Wireless Personal Communications*, 117:1–22, 04 2021.

[114] Mohammad Ehteram, Akram Seifi, and Fatemeh Barzegari Banadkooki. *Sunflower Optimization Algorithm*, pages 43–47. Springer Nature Singapore, Singapore, 2023.

[115] Shuang Wang. Multipath routing based on genetic algorithm in wireless sensor networks. *Mathematical Problems in Engineering*, 2021:1–6, 06 2021.

[116] Saeid Jazebi and Ali Ghaffari. Risa: routing scheme for internet of things using shuffled frog leaping optimization algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 11, 10 2020.

[117] Bestan B Maaroof, Tarik A Rashid, Jaza M Abdulla, Bryar A Hassan, Abeer Alsadoon, Mokhtar Mohammadi, Mohammad Khishe, and Seyedali Mirjalili. Current studies and applications of shuffled frog leaping algorithm: a review. *Archives of Computational Methods in Engineering*, 29(5):3459–3474, 2022.

[118] Neha Sharma, Usha Batra, and Sherin Zafar. Remit accretion in iot networks encircling ingenious firefly algorithm correlating water drop algorithm. *Procedia Computer Science*, 167:551–561, 2020. International Conference on Computational Intelligence and Data Science.

[119] Basem O. Alijla, Li-Pei Wong, Chee Peng Lim, Ahamad Tajudin Khader, and Mohammed Azmi Al-Betar. A modified intelligent water drops algorithm and its application to optimization problems. *Expert Systems with Applications*, 41(15):6555–6569, 2014.

[120] Jun Li, Xiaoyu Wei, Bo Li, and Zhigao Zeng. A survey on firefly algorithms. *Neurocomputing*, 500:662–678, 2022.

[121] Jonny Karlsson, Laurence S. Dooley, and Göran Pulkkis. Secure routing for manet connected internet of things systems. In *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 114–119, 2018.

[122] Xiaoqiang Zhao, Shaoya Ren, Heng Quan, and Qiang Gao. Routing protocol for heterogeneous wireless sensor networks based on a modified grey wolf optimizer. *Sensors*, 20(3), 2020.

[123] Shailja Agnihotri and K. R. Ramkumar. Content based routing algorithm to improve qos in iomt networks. In Ashim Saha, Nirmalya Kar, and Suman Deb, editors, *Advances in Computational Intelligence, Security and Internet of Things*, pages 195–206, Singapore, 2020. Springer Singapore.

[124] Heyan Zhang. Secure routing protocol using salp-particle swarm optimization algorithm. *Journal of Networking and Communication Systems (JNACS)*, 3:1–10, 07 2020.

[125] Mauro Castelli, Luca Manzoni, Luca Mariot, Marco S. Nobile, and Andrea Tangherloni. Salp swarm optimization: A critical review. *Expert Systems with Applications*, 189:116029, 2022.

[126] Alireza Ebrahimi Basabi, Jingsha He, and Seyed Mahmood Hashemi. Secure routing in iot with evolutionary algorithm. *Journal of Advances in Computer Networks*, 7(2), 2019.

[127] Mohammad Shahrazad and Amirhessam Alikhanzadeh. Application of imperialist competitive optimization algorithm in power industry. *Journal of Industrial Engineering Computations*, 6:43–58, 12 2015.

[128] Shu-Hung Lee, Chia-Hsin Cheng, Chien-Chih Lin, and Yung-Fa Huang. Pso-based target localization and tracking in wireless sensor networks. *Electronics*, 12(4), 2023.

[129] Farzad Kiani and Amir Seyyedabbasi. *Metaheuristic Algorithms in IoT: Optimized Edge Node Localization*, pages 19–39. Springer International Publishing, Cham, 2023.

[130] Saroj Sahoo, Apu Kumar Saha, Absalom Ezugwu, Ovre Agushaka, Belal Abuhaija, Anas Alsoud, and Laith Abualigah. Moth flame optimization: Theory, modifications, hybridizations, and applications. *Archives of Computational Methods in Engineering*, 30, 08 2022.

[131] Saeid Barshandeh, Mohammad Masdari, Gaurav Dhiman, Vahid Hosseini, and Krishna K Singh. A range-free localization algorithm for iot networks. *International Journal of Intelligent Systems*, 37(12):10336–10379, 2022.

[132] Satnam Kaur, Lalit K. Awasthi, A.L. Sangal, and Gaurav Dhiman. Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Engineering Applications of Artificial Intelligence*, 90:103541, 2020.

[133] B.K. Tripathy, Praveen Reddy, Viet Pham, Thippa Gadekallu, Kapal Dev, Sharnil Pandya, and Basem ElHalawany. Harris hawk optimization: A survey on variants and applications. *Computational Intelligence and Neuroscience*, 2022, 06 2022.

[134] He Huang, Jianfei Yang, Xu Fang, Hao Jiang, and Lihua Xie. An improved pso approach to indoor localization system based on imu, wifi rss and map information. In *2022 IEEE 17th International Conference on Control and Automation (ICCA)*, pages 692–697, 2022.

[135] Pudi Sekhar, E. Laxmi Lydia, Mohamed Elhoseny, Marwan Al-Akaidi, Mahmoud M. Selim, and K. Shankar. An effective metaheuristic based node localization technique for wireless sensor networks enabled indoor communication. *Physical Communication*, 48:101411, 2021.

[136] Yiying Zhang and Zhigang Jin. Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems. *Expert Systems with Applications*, 148:113246, 2020.

[137] Sheetal N. Ghorpade, Marco Zennaro, and Bharat S. Chaudhari. Gwo model for optimal localization of iot-enabled sensor nodes in smart parking systems. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):1217–1224, 2021.

[138] Sihem Tlili, Sami Mnasri, and Thierry Val. A multi-objective gray wolf algorithm for routing in iot collection networks with real experiments. In *2021 National Computing Colleges Conference (NCCC)*, pages 1–5, 2021.

[139] Manal Tamir, Raddouane Chiheb, Fatima Ouzayd, and Kawtar Retmi. Conception of an automatic decision support platform based on cross-sorting methods and the fuzzy logic for general use. In Mohamed Lazaar, Claude Duvallet, Abdellah Touhafi, and Mohammed Al Achhab, editors, *Proceedings of the 5th International Conference on Big Data and Internet of Things*, pages 73–81, Cham, 2022. Springer International Publishing.

[140] Adrien van den Bossche, Rejane Dalce, and Thierry Val. locura4iot-a testbed dedicated to accurate localization of wireless nodes in the iot. *IEEE Sensors Journal*, 12 2021.

[141] Locura4iot dataset. `http://locura4iot.irit.fr/datasets/20220611-uwb-twr/`, 2022. Accessed: 2023-02-20.

[142] Sihem Tlili, Mnasri Sami, and Thierry Val. Uwb time of flight-based indoor iot localisation solution with deep learning optimised by meta-heuristics. *International Journal of Sensor Networks*, 44:99–123, 01 2024.

[143] M5stickc. `https://docs.m5stack.com/en/core/m5stickc`, 2024. Accessed: 2024-04-07.

[144] Painless mesh library on arduino. `https://www.arduino.cc/reference/en/libraries/painless-mesh/`, 2024. Accessed: 2024-04-07.

[145] Espnow. `https://www.espressif.com/en/products/software/esp-now/overview`, 2024. Accessed: 2024-05-01.

[146] Wajih Abdallah. *La résolution du déploiement 3D d'objets connectés sans fil à l'intérieur en utilisant un schéma hybride entre les méthodes géométriques de déploiement et les algorithmes d'optimisation distribués*. Theses, Université Toulouse le Mirail - Toulouse II, July 2022.

[147] Antonio J. Nebro. Github - jmetal/jmetal: jmetal: a framework for multi-objective optimization with metaheuristics.

[148] Institut de recherche en informatique de toulouse. https://www.irit.fr/en/home/, 2024. Accessed: 2024-04-07.

[149] Joaquin Derrac, Salvador Garcia, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[150] V. Roshan Joseph. Optimal ratio for data splitting. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(4):531–538, 2022.

[151] Qi Wang, Yue Ma, Kun Zhao, and Yingjie Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 9, 04 2022.

[152] Francois Chollet et al. Keras. https://github.com/fchollet/keras, 2023. Accessed: 2023-11-07.

[153] Emmanuel Dada, Stephen Joseph, David Oyewola, Alaba Fadele, Haruna Chiroma, and Shafi i Abdulhamid. Application of grey wolf optimization algorithm: Recent trends, issues, and possible horizons. *GAZI UNIVERSITY JOURNAL OF SCIENCE*, 35:485–504, 03 2022.

[154] Bing Jia, Zhaopeng Zong, Baoqi Huang, and Thar Baker. *A DNN-based WiFi-RSSI Indoor Localization Method in IoT*, pages 200–211. 02 2021.

[155] Chao Cheng and Zhi-yang Jiang. A weighted centroid localization algorithm based on rssi adaptive value coupled with two norm improvements. 01 2018.

[156] Abdelmoumen Norrdine. An algebraic solution to the multilateration problem. 04 2015.

[157] Ahcène Bounceur, Olivier Marc, Massinissa Lounis, Julien Soler, Laurent Clavier, Pierre Combeau, Rodolphe Vauzelle, Loïc Lagadec, Reinhardt Euler, Madani Bezoui, and Pietro Manzoni. Cupcarbon-lab: An iot emulator. pages 1–2, 01 2018.

[158] Cupcarbon digital twin iot. https://cupcarbon.com/, 2024. Accessed: 2024-04-07.

[159] Mqtt: The standard for iot messaging. https://mqtt.org/, 2024. Accessed: 2024-05-01.

[160] Apache kafka. https://kafka.apache.org/, 2024. Accessed: 2024-04-24.

[161] Apache flink. https://flink.apache.org/, 2024. Accessed: 2024-04-24.

Appendix A

Résumé Long en Français

Routage et localisation en intérieur dans les réseaux IoT : approches intelligentes basées sur les métaheuristiques et l'apprentissage profond

Mots-clés : Internet des objets, Intelligence Artificielle, Optimisation, Métaheuristiques, Apprentissage profond, Multi-objectifs, Routage, Localisation en intérieur, Réseaux IoT, Réseaux maillés, Gray Wolf Optimizer, Edge Computing.

A.1 Motivations et problématique

L'Internet des Objets (Internet of Things-IoT) a connu une expansion importante au cours des dernières années, s'imposant comme un pilier majeur de notre vie moderne. Des applications diverses et variées ont émergé. Elles révolutionnent notre façon d'interagir avec le monde qui nous entoure. En offrant diverses possibilités, l'IoT a transformé de nombreux aspects de la vie quotidienne, avec des applications qui s'intègrent dans divers domaines.
Cependant, cette expansion exponentielle de l'IoT a créé certains défis. Parmi ceux-ci, le problème du routage dans les réseaux IoT à topologie maillée se pose comme un défi majeur à résoudre, notamment en raison de leur nature dynamique et de leur évolutivité topologique constante. Le routage consiste à sélectionner les chemins parmi ceux qui sont possibles pour le transfert de données entre les différents appareils connectés au sein du réseau. Un routage efficace dans les réseaux IoT est essentiel pour garantir des communications fluides, fiables et économes en énergie. Tout d'abord, il permet d'optimiser les performances en réduisant les retards de transmission et en maximisant l'utilisation des ressources réseau. De plus, un routage efficace favorise la fiabilité de la communication en choisissant des chemins fiables, minimisant ainsi les risques de perte de données ou d'interruptions de service au sein du réseau IoT. En outre, il contribue à une économie d'énergie, notamment dans les dispositifs alimentés par batterie. À cette fin, il limite les transmissions inutiles et privilégie des chemins économes en énergie. De plus, un routage efficace permet au réseau IoT

de s'adapter de manière dynamique aux changements de topologie et d'évoluer pour intégrer de nouveaux appareils. Cette adaptation garantit la flexibilité et la scalabilité du réseau. Il joue également un rôle crucial dans la sécurisation des données et la protection des informations sensibles transitant dans le réseau IoT. Dans ce contexte, des stratégies de routage sécurisées doivent être appliquées pour prévenir les attaques. Ainsi, un routage efficace constitue un pilier essentiel pour assurer le bon fonctionnement des applications IoT dans divers domaines.
Un autre défi crucial dans les réseaux IoT est celui de la localisation en intérieur. Elle consiste à déterminer la position physique des appareils connectés à l'intérieur d'un espace d'intérieur clos. Contrairement aux systèmes de localisation en extérieur, les techniques traditionnelles basées sur le GNSS rencontrent des difficultés pour fournir des solutions précises et fiables dans un environnement interne et complexe. Une localisation précise en intérieur dans les réseaux IoT est essentielle pour améliorer la qualité des services proposés en offrant plusieurs avantages clés. Tout d'abord, elle optimise la gestion des objets connectés en permettant un suivi efficace des équipements et des produits. En outre, elle renforce la sécurité en facilitant la surveillance des zones sensibles et la détection d'intrusions. De plus, elle facilite la navigation des utilisateurs dans des environnements complexes, ce qui améliore l'expérience utilisateur. Elle permet également une personnalisation des services basée sur la localisation, répondant ainsi aux besoins spécifiques spatiaux des utilisateurs. Elle facilite aussi l'automatisation des tâches en identifiant la position des objets ou des personnes en temps réel. En résumé, une localisation précise en intérieur dans les réseaux IoT est essentielle pour optimiser les opérations et pour améliorer la qualité des services proposés. Face à ces défis majeurs, une résolution efficace des problèmes de routage et de localisation en intérieur dans les réseaux IoT devient impérative. Cependant, la complexité et la dynamique de ces réseaux exigent l'application de méthodologies puissantes et innovantes.
C'est dans ce contexte que l'optimisation multi-objectifs se révèle être une approche prometteuse. Cette méthode offre la possibilité de prendre en compte simultanément plusieurs critères de performance. Elle permet ainsi de trouver des solutions optimales dans des environnements aussi complexes que les réseaux IoT. De plus, l'utilisation de métaheuristiques s'avère être particulièrement pertinente pour résoudre efficacement les problèmes réels auxquels nous sommes confrontés. Parallèlement, les techniques de l'apprentissage profond (DL) ont démontré leur capacité à aborder de manière efficace les problèmes complexes. Elles offrent des solutions innovantes et adaptatives. Grâce à leur capacité à extraire des modèles à partir de données, les techniques de Deep Learning ouvre de nouvelles perspectives pour la résolution des défis posés par les réseaux IoT.
C'est dans ce contexte que s'intègrent les contributions de cette thèse. La manière dont le manuscrit est structuré est décrite dans la section suivante.

A.2 Organisation du manuscrit

Ce document est structuré en quatre chapitres organisés en deux parties distinctes. La première partie présente l'état de l'art dans les chapitres 2 et 3. Ensuite, la seconde partie détaille les contributions proposées dans les chapitres 4 et 5.
Dans le chapitre 2, nous explorons le routage et la localisation en intérieur dans le contexte des réseaux IoT. Nous commençons par une introduction globale à l'IoT. Elle présente ses différentes topologies de réseau, ses technologies de communication, ainsi que ses divers domaines d'application et les principaux défis auxquels il est confronté. Ensuite, nous définissons le concept

du routage dans les réseaux IoT, ses divers types, ainsi que les indicateurs de performance utilisés pour évaluer les algorithmes de routage. À la suite, nous synthétisons les recherches récentes sur le routage dans les réseaux IoT. Nous visons à développer un routage géographique qui tire avantage des positions des noeuds du réseau pour optimiser le processus de routage. À cette fin, nous examinons ensuite la localisation en intérieur dans les réseaux IoT. Nous débutons par sa définition et la description de son importance. Par la suite, nous examinons les techniques appliquées pour créer des solutions de localisation en intérieur, ainsi que les critères d'évaluation associés à ces solutions. Enfin, nous comparons et analysons des études récentes sur la localisation en intérieur dans les réseaux IoT.

Le chapitre 3 est dédié à l'exploration approfondie de deux domaines prometteurs : l'optimisation et les techniques de Deep Learning. La première partie expose les différents types de problèmes d'optimisation, leurs concepts fondamentaux, ainsi que les principales méthodes de résolution et les critères d'évaluation qui leur sont associés. Elle se termine par une revue des méthodes récentes utilisées pour résoudre ces problèmes d'optimisation. La seconde partie de ce chapitre s'intéresse au DL. Nous présentons ses principaux domaines d'application, notamment son utilisation dans le cadre de la régression. Nous abordons également ses fondements théoriques, les principales techniques de DL applicables aux problèmes de régression, ainsi que les critères de performance utilisés pour les évaluer. À la suite, ce chapitre propose une revue détaillée comparant les études récentes sur la résolution du problème du routage dans les réseaux IoT basées sur l'utilisation d'algorithmes d'optimisation. Enfin, il examine les recherches récentes portant sur la résolution du problème de localisation en intérieur dans les réseaux IoT en se basant sur l'utilisation des métaheuristiques et/ou de techniques de DL.

Le chapitre 4 propose une description approfondie de nos contributions théoriques. Il débute par une modélisation mathématique du routage dans les réseaux IoT. Ensuite, nous présentons une amélioration de l'algorithme métaheuristique d'optimisation MOGWO existant, nommée Improved MOGWO (IMOGWO). Par la suite, nous détaillons une approche d'optimisation des paramètres des modèles DL en utilisant les métaheuristiques. Dans la section suivante, nous décrivons notre proposition de localisation en intérieur dans les réseaux IoT fondée sur l'approche précédemment exposée. Cette solution de localisation va nous permettre de développer un algorithme de routage géographique et hybride. Il est conçu spécifiquement pour les réseaux IoT à architecture maillée. La dernière section de ce chapitre est dédiée à la présentation détaillée de cette proposition de routage.

Le dernier chapitre, 5, se concentre sur la présentation des expérimentations et des simulations réalisées pour évaluer nos contributions, ainsi qu'à l'analyse des résultats obtenus. Nous évaluons notre proposition de routage réactif et multi-objectifs basée sur MOGWO à travers des expériences réelles détaillées et présentées dans la première section. Les résultats obtenus sont également exposés et analysés pour mettre en évidence les performances de l'approche proposée. Par la suite, l'évaluation théorique des performances de l'algorithme IMOGWO proposé en résolvant les problèmes de référence DTLZ [4] est présenté dans la deuxième section. Les résultats obtenus par IMOGWO ainsi que par d'autres algorithmes d'optimisation largement utilisés sont analysés et comparés. Dans la troisième section, nous détaillons les tests menés pour évaluer l'efficacité de nos contributions : (i) l'optimisation des poids des modèles DL par des algorithmes métaheuristiques et (ii) le développement d'un modèle DL de localisation en intérieur basé sur les mesures UWB ToF. Les résultats de l'évaluation de notre approche sont comparés et analysés par rapport à d'autres solutions de localisation en intérieur. Dans la dernière section, nous présentons et analysons les

résultats des expérimentations menées sur notre proposition de routage géographique et hybride, qui repose sur IMOGWO. Ces tests visent trois objectifs : (i) évaluer les performances de notre solution de routage proposée, (ii) les comparer à celles d'autres solutions de routage existantes, et (iii) vérifier l'efficacité de l'algorithme IMOGWO dans un contexte d'un problème réel, en dehors des problèmes de référence DTLZ.
Pour terminer, nous exposons nos conclusions et nous abordons également nos perspectives ainsi que les directions futures de notre recherche.

A.3 Étude bibliographique et principales contributions

Le but de cette thèse est de combiner les métaheuristiques et l'apprentissage profond pour proposer des approches pour le routage dans les réseaux IoT maillés et pour la localisation en intérieur.

A.3.1 Étude bibliographique

En première étape, nous avons effectué une revue approfondie de la littérature. L'état de l'art porte sur les sujets abordés : le routage et la localisation en intérieur dans un réseau IoT, l'optimisation métaheuristique et le Deep Learning. Nos publications [8] et [7] s'inscrivent dans ce contexte.

A.3.1.1 Travaux en relation avec le problème de routage dans les réseaux IoT

Dans la littérature, diverses études se concentrent sur la résolution du problème du routage dans les réseaux IoT. Les auteurs de [11] proposent un algorithme de routage hiérarchique dont l'objectif est d'assurer une efficacité énergétique et une sécurité dans une topologie dynamique. La proposition réduit la consommation d'énergie et offre un pourcentage de sécurité de 93%. Cependant, la validation de la proposition est basée uniquement sur des tests Matlab, sans scénario de simulation IoT ou d'expérimentation du monde réel. De plus, le réseau simulé est limité à 5 noeuds. Un routage basé sur la négociation est proposé par [13] en se basant sur l'apprentissage par renforcement multi-agent. Les résultats obtenus montrent une réduction du temps d'exécution de 88% et une précision moyenne de 77%. L'inconvénient de ce travail est que l'algorithme proposé n'est pas évalué selon les métriques connues pour le routage et n'est pas comparé avec d'autres algorithmes de routage. Un routage basé sur la QoS est proposé dans [15]. L'objectif est de minimiser la consommation d'énergie et améliorer la sécurité du routage en appliquant la métaheuristique ACO. L'algorithme de routage introduit améliore la consommation d'énergie de plus de 9.31% par rapport aux autres algorithmes comparés. En revanche, le modèle du réseau étudié n'inclut ni de noeuds mobiles ni de noeuds hétérogènes. De plus, les résulats ne sont pas validés par des expérimentations réelles.
À partir de la synthèse de ces travaux et d'autres parmi ceux qui sont les plus récents, nous avons constaté que chacun d'eux se concentre sur un nombre restreint de besoins et de critères. De plus, la plupart de ces travaux ciblent un type spécifique d'environnement IoT, ce qui limite leur efficacité pour relever les défis rencontrés dans des autres contextes IoT. En outre, la plupart des études se basent sur des simulations pour illustrer les améliorations de performances, mais elles ne sont pas réalisées à l'aide d'un simulateur spécifique aux réseaux IoT, et encore moins sur des testbeds réels. Par conséquent, nous envisageons de proposer un algorithme de routage qui prenne en compte un plus large éventail d'objectifs, qui peut s'adapter plus généralement aux environnements IoT, et

qui améliore les conditions d'expérimentation et d'évaluation de la solution de routage. Parmi les divers types de routage, nous nous intéressons particulièrement au routage géographique en raison de ses avantages et de sa capacité à constituer une solution de routage attrayante pour les réseaux IoT. [9] et [10] ont mené des études sur les protocoles de ce type. Ces études recommandent l'utilisation du routage géographique comme une solution clé pour la transmission d'informations, en surmontant les défis liés à l'évolutivité et à la mobilité.

A.3.1.2 Travaux en relation avec le problème de localisation en intérieur dans les réseaux IoT

Dans la littérature, plusieurs études se focalisent sur la résolution du problème de localisation en intérieur dans les réseaux IoT. Les auteurs de [43] développent WRCDV-Hop qui comporte quatre améliorations de l'algorithme DV-HoP. L'algorithme proposé se repose sur la métaheuristique "Whale Optimization Algorithm" (WOA). La précision de localisation obtenue par WRCDV-Hop est de 98.8%. Cependant, les résultats sont basés sur des simulations Matlab plutôt que sur des données réelles. CSILoc est un framework développé et proposé par [46]. Il inclut une solution DL visant à obtenir une estimation précise de la localisation. En revanche, les résultats obtenus ne répondent pas aux exigences de localisation intérieure de l'IoT puisque le MAE obtenu est de 1.75m et sa précision de localisation est de 68.5%. Une autre approche de localisation est proposée apr [49]. Elle utilise une image 2D pour servir de base à un modèle DL. L'approche applique une variante améliorée de l'algorithme PSO afin d'optimiser l'entraînement du modèle DL utilisé. Cependant, la précision obtenue (97.92%) est fondée sur les résultats d'une phase de validation. Toutefois, la fiabilité de la proposition doit être vérifiée à travers une étape de test distincte pour confirmer leur validité. L'examen des travaux susmentionnés ainsi que d'autres études proposant des solutions de localisation en intérieur montre que les méthodes actuelles ne parviennent pas, généralement, à atteindre la précision de localisation requise dans les environnements IoT. En effet, une précision sub-métrique est exigée pour les systèmes de localisation en intérieur dans ces environnemets [42]. Cependant, les recherches ayant atteint des erreurs moyennes inférieures à 1m ont obtenu ces résultats à partir de simulations plutôt que de données réelles. De plus, ces simulations n'ont pas été réalisées à l'aide d'un simulateur dédié aux réseaux IoT prenant en compte les liens radio spécifiques et leurs technologies de transmissions particulières. Dans ce cadre, nous pensons à proposer une solution de localisation intérieure précise, conforme aux exigences des environnements IoT, en se basant sur des données réelles.

A.3.1.3 Travaux en relation avec les méthodes récentes de résolution des problèmes d'optimisation

Les recherches visant à améliorer les performances de résolution des problèmes d'optimisation sont en constante évolution. Dans ce contexte, le EMOPSOC est un algorithme hybride multi-objectifs poposé par [75]. Il est développé pour résoudre le problème de planification et d'allocation des ressources dans les environnements de fog computing. L'algorithme proposé offre une augmentation de 80% pour les deux indicateurs Hypervolume et le IGD. Cependant, EMOPSOC est spécifique à un type particulier de problèmes. De plus, il a été testé avec seulement 3 objectifs. Il est également caractérisé par une complexité computationnelle accrue. Les auteurs de [78] proposent une métaheuristique afin de réaliser des optimisations à 2 objectifs de conceptions de bâtiments réels. En plus de son coût computationnel élevé, l'algorithme proposé nécessite un ajustement des paramètres pour des performances optimales. En outre, l'applicabilité de la proposition à d'autres types de problèmes nécessite d'être vérifiée. Une version améliorée de l'algorithme PSO, nommée QSPSO a été proposée dans [82]. Les performances de QSPSO sont testés en résolvant les fonctions de test de référence (F1-F21). Son inconvénient est qu'il utilise plusieurs paramètres qui doivent être réglés, en plus de son coût computationnel élevé en raison des calculs supplémentaires pour les composants de la mécanique quantique et des solitons. De plus, QSPSO nécessite d'être validé en le testant avec des problèmes réels. L'étude de ces travaux et d'autres travaux récents visant à améliorer la résolution des problèmes d'optimisation met en évidence certains inconvénients. Tout d'abord, de nombreuses méthodes de résolution sont conçues pour des types spécifiques de problèmes, ce qui nécessite de démontrer leur efficacité et leurs bonnes performances sur une gamme plus large de problèmes d'optimisation. De plus, certaines méthodes proposées ne sont testées que sur des problèmes de référence, ne garantissant pas leur efficacité dans des scénarios réels. En outre, la plupart des propositions sont limitées en termes du nombre d'objectifs qu'elles peuvent traiter efficacement. Les bonnes performances sont généralement obtenues uniquement pour un nombre restreint d'objectifs. L'adaptation des paramètres est également cruciale pour de nombreux algorithmes de résolution, ce qui peut impacter significativement la qualité des résultats obtenus. Ainsi, il est essentiel de poursuivre les efforts de recherche pour développer des méthodes de résolution offrant à la fois des performances élevées sur des problèmes de référence et des problèmes réels, tout en pouvant gérer un nombre important d'objectifs.

A.3.1.4 Travaux en relation avec l'optimisation des techniques DL par métaheuristiques

Dans le domaine de la recherche, une tendance émerge visant à optimiser les techniques de Deep Learning à l'aide de métaheuristiques. [97] utilise le "Particle Swarm Optimization without Velocity" (PSWV) pour optimiser les hyperparamètres des CNNs. La proposition a été testée pour résoudre un problème de classification d'images. Le taux d'erreurs obtenu est de 3.64%. La proposition de [98] optimise les hyperparamètres des LSTMs en appliquant le "Artificial Rabbits Optimization Algorithm" (ARO) [99]. Les résultats de prédiction du prix des actions donne un MSE de 24.287 et un MAE de 3.804. Les auteurs de [100] appliquent IPSO, une version améliorée de PSO, pour optimiser les hyperparamètres des DNNs. En évaluant l'impact de la distanciation sociale sur la propagation du COVID-19 par la proposition, les résultats donne un MAE de 4.8177 et un RMSE de 45.0471. L'étude de ces travaux et d'autres parmi ceux qui sont les plus récents montre qu'ils présentent certaines limites. Certains travaux se concentrent sur l'utilisation de métaheuristiques pour optimiser les hyperparamètres des modèles DL, tandis que d'autres visent à optimiser à la fois les hyperparamètres et les valeurs initiales des poids de ces modèles. D'autres cherchent à optimiser les données elles-mêmes en réduisant leur taille et en sélectionnant les attributs les plus pertinents grâce à des métaheuristiques. Cependant, une limitation de ces approches est qu'elles sont souvent conçues pour des types spécifiques de techniques de DL, manquant ainsi de généralité. Bien que l'utilisation de métaheuristiques ait prouvé son efficacité pour améliorer les performances des modèles DL étudiés, l'étude montre qu'il n'existe pas de travaux exploitant les métaheuristiques pour optimiser les poids des modèles DL pendant l'entraînement. Il est donc nécessaire de réfléchir à l'optimisation des poids par métaheuristiques afin de tirer pleinement parti de leurs avantages, accélérant ainsi la convergence vers des valeurs optimales des poids et améliorant les performances des modèles entraînés. De plus, il est crucial que les optimiseurs proposés soient généraux et applicables à plusieurs techniques de DL plutôt que spécifiques à un seul type.

A.3.1.5 Travaux en relation avec la résolution du problème de routage dans les réseaux IoT en se basant sur les algorithmes d'optimisation

De nombreux travaux de recherche visent actuellement à améliorer les performances du routage dans les réseaux IoT en utilisant des métaheuristiques. Dans ce contexte, [109] propose un routage géographique. Les routes sont sélectionnées en prenant en compte la position des noeuds, leur niveau d'énergie et les données sur le trafic. la métaheuristique utilisée est le BFO [110]. Les résultats montrent une réduction de délai de transmission de 12% et une augmentation de PDR de 7 à 9%. Cependant, le modèle du réseau étudié ne prend pas en compte les noeuds mobiles. De plus, l'algorithme suppose que les noeuds connaissent leur propre emplacement ainsi que celui de leurs voisins, ce qui n'est toujours vrai dans des scénarios réels. En outre, les simulations ont été effectuées avec Matlab, plutôt qu'un simulateur spécifique aux réseaux IoT. En se basant sur une nouvelle variante de PSO proposée, [112] propose un algorithme de routage basé sur la QoS. Les résultats obtenus montrent que la durée de vie du réseau est augmentée de 14% à 29%. de plus, la consommation d'énergie est réduite de 61% à 70%. Cependant, ces résultats sont obtenus en comparant l'algorithme proposé avec d'autres algorithmes d'optimisation plutôt qu'avec d'autres solutions de routage existantes. Les auteurs de [113] proposent un algorithme de routage multi-chemins. Ils appliquent un nouvel algorithme d'optimisation hybride en intégrant le SFO [114] et le GWO. Les résultats des simulations effectuées montrent un délai de transmission moyen de 0.779 s et un débit amélioré de 47.368%. En revanche le processus de routage développé prend en compte une seule fonction objectif. L'examen des travaux antérieurs ainsi que d'autres études récentes met en évidence l'apport significatif des métaheuristiques dans l'amélioration des performances de routage dans les réseaux IoT. Cependant, malgré ces progrès, le routage dans les réseaux IoT présente encore de nombreux défis, capables d'impacter la performance globale du réseau. Chacun des protocoles étudiés se concentre sur un nombre limité de besoins, ce qui signifie qu'aucun d'entre eux n'est entièrement efficace pour relever tous les défis. De plus, ces travaux présentent des inconvénients, notamment le fait que les expériences se limitent souvent à des simulations, et la plupart des propositions n'utilisent pas de simulateurs dédiés aux réseaux IoT, ce qui pourrait affecter la validité des résultats. De plus, ces travaux ont souvent un nombre limité d'objectifs. Cependant, dans un contexte où les applications IoT exigent toujours plus de performances et de vitesse dans les communications, en particulier pour les applications en temps réel, il est impératif de concevoir un protocole plus robuste et efficace pour l'IoT. Il a été démontré que l'utilisation des algorithmes d'optimisation améliore les performances et l'efficacité des stratégies de routage dans les réseaux IoT. Ainsi, il est crucial de profiter de leur succès et leur capacité à résoudre des problèmes multi-objectifs complexes pour développer de nouvelles solutions de routage visant à obtenir des performances accrues dans les réseaux IoT.

A.3.1.6 Travaux en relation avec la résolution du problème de localisation en intérieur dans les réseaux IoT en se basant sur les métaheuristiques et/ou les techniques DL

Dans la littérature, plusieurs recherches sont menées en utilisant les métaheuristiques et les techniques DL pour développer des solutions de localisation en intérieur dans les réseaux IoT. [44]

propose une optimisation de l'algorithme de localisation APIT en utilisant les ANNs. L'erreur de distance moyenne obtenue est de 1.55m. Elle ne satisfait pas aux exigences des applications IoT. De plus, les résultats sont issus des simulations effectuées à l'aide de Matlab au lieu d'un simulateur dédié aux réseaux IoT. Les auteurs de [131] proposent un algorithme d'optimisation hybride en combinantle "Tunicate Swarm Algorithm" [132] et le "Harris hawk optimization" [133]. Ensuite, l'algorithme hybride a été intégré à l'algorithme DV-HoP. Les résulats des tests Matlab donnent une erreur de distance moyenne de 0.45 m. Cependant, l'hybridation de deux algorithmes métaheuristiques peut augmenter la complexité de l'algorithme de localisation. De plus, le réglage des paramètres peut s'avérer difficile et nécessite plus du temps. [134] développe une approche hybride combinant les points forts d'une version améliorée de PSO, le "Inertial Measurement Unit" (IMU) et le RSSI fingerprinting. L'erreur de distance moyenne obtenue est de 0.7m. En revanche, la combinaison de 3 méthodes augmente la complexité de l'algorithme de localisation proposé. Un autre inconvénient de la proposition est qu'elle utilise des mesures RSSI qui peuvent être sensibles et affectées par des facteurs environnementaux. De plus, la solution nécessite des informations cartographiques précises ce qui peut être difficile et peut prendre plus du temps. D'autres récentes recherches explorent également l'utilisation de métaheuristiques et/ou de techniques Deep Learning pour développer des solutions de localisation en intérieur dans les réseaux IoT. L'étude de ces travaux se distinguent par leur approche méthodologique : certains s'appuient uniquement sur les métaheuristiques, d'autres sur les techniques de DL, tandis que d'autres combinent les deux paradigmes de manière hybride. Cette diversité d'approches démontre le potentiel des métaheuristiques et des techniques de DL pour résoudre les défis complexes de localisation en intérieur et pour améliorer les résultats dans ce domaine spécifique. Cependant, ces approches présentent des inconvénients notables. Tout d'abord, la plupart des travaux se basent sur des expériences simulées plutôt que sur des données réelles, limitant ainsi la validité des résultats obtenus. De plus, l'utilisation généralisée des mesures RSSI, sensibles aux perturbations environnementales telles que les interférences et l'atténuation du signal, constitue un défi majeur. Malgré les avantages et l'efficacité des approches basées sur les métaheuristiques et le DL, les résultats obtenus restent souvent insuffisants pour répondre aux exigences croissantes des réseaux IoT en termes de performances et de précision de localisation. Ainsi, il est impératif d'explorer de nouvelles approches permettant de tirer pleinement parti des avantages combinés de ces deux paradigmes prometteurs tout en utilisant des mesures moins sensibles que le RSSI.

A.3.2 Modélisation mathématique

En deuxième étape, une modélisation mathématique du routage dans les réseaux IoT est proposée. Elle identifie et formalise les caractéristiques du réseau IoT à prendre en compte, ainsi que les objectifs clés, les variables de décision et les contraintes pour le problème de routage. Cette modélisation offre un cadre pour la conception et l'optimisation des algorithmes de routage dans les réseaux IoT. Elle rend les approches de routage proposées plus structurées, ce qui simplifie la compréhension de leur fonctionnement et de leur évaluation. Elle prend en considération des objectifs essentiels tels que la progression positive vers la destination finale, la gestion efficace de l'énergie, le maintien des liaisons fiables entre les objets, la garantie de la continuité des routes et la minimisation de la latence de transmission. En mettant l'accent sur ces objectifs, cette modélisation vise à fournir une base solide pour le développement de solutions de routage robustes et efficaces.

A.3.3 Un routage réactif et multi-objectifs basé sur MOGWO

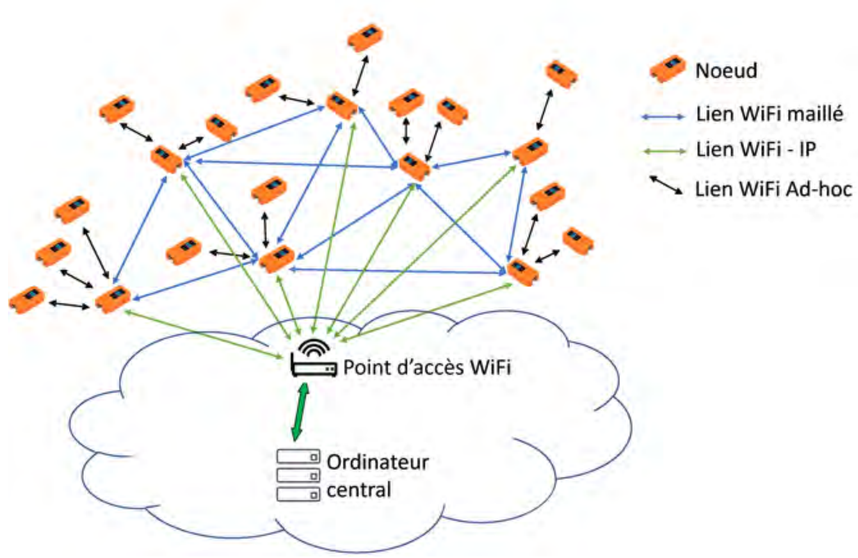
Dans une deuxième contribution, un routage réactif et multi-objectifs est proposé. L'approche proposée consiste à l'exploitation de l'algorithme MOGWO [2] afin de modéliser un routage multi-sauts dans un réseau IoT à topologie hybride. MOGWO est une méta-heuristique récente qui a gagné un intérêt croissant dans divers domaines. Dans cette proposition, le routage est abordé comme un problème d'optimisation où les chefs d'étoiles représentent les solutions dans l'espace de recherche. L'objectif est de définir une route efficace à travers des sauts multiples entre ces chefs d'étoiles. En utilisant l'algorithme MOGWO, la solution optimale est déterminée, offrant ainsi la route à suivre pour l'acheminement des données. Le modèle de réseau IoT que nous envisageons repose sur une topologie hybride, ce qui implique une structuration en étoiles. Chaque étoile est constituée d'un appareil spécifique, désigné comme son chef, jouant ainsi le rôle de point central pour la gestion du trafic au sein de l'étoile. Notre contribution dans cette section se concentre sur le routage multi-objectifs au sein du réseau IoT à topologie hybride. Elle se focalise spécifiquement sur le processus de routage lui-même, excluant ainsi la formation des étoiles et la sélection de leurs chefs. Nous supposons donc que le modèle du réseau ainsi que ses composantes, telles que les étoiles et leurs chefs, sont préalablement établis. Les chefs des étoiles maintiennent une communication avec un ordinateur central via des chemins préétablis, étant donné que ces chefs sont stationnaires et ne changent pas de position. Ils envoient régulièrement des messages de contrôle à cet ordinateur central, contenant des données sur leurs niveaux d'énergie restante ainsi que sur la composition des étoiles, en cas de changement. L'ordinateur central a une vision globale du réseau ainsi que des liens entre les chefs des étoiles. Il utilise les informations reçues périodiquement de la part des chefs des étoiles pour actualiser sa connaissance globale du réseau. Son rôle consiste à appliquer l'algorithme métaheuristique, utilisant ces informations actualisées de manière périodique, pour élaborer les routes de communication par un routage réactif et multi-objectifs. Cette approche intègre également deux objectifs essentiels : la gestion efficace de l'énergie et la réduction de la latence de transmission. Lorsqu'un objet de l'étoile souhaite transmettre un message, il le fait parvenir simplement à son chef d'étoile, qui se charge alors de demander à l'ordinateur central d'établir une route optimisée vers le destinataire final. Ensuite, l'ordinateur central initie un processus de routage multi-sauts entre les différents chefs d'étoiles du réseau basé sur l'optimiseur multi-objectifs MOGWO. La route multi-sauts commence par le chef d'étoile contenant la source initiale et se poursuit jusqu'au chef de l'étoile contenant la destination finale, si celui-ci est situé dans l'environnement IoT. Dans le cas contraire, si la destination se trouve à l'extérieur du réseau IoT, la route se poursuit jusqu'à atteindre une passerelle appropriée pour être transmise vers l'extérieur.

Expérimentations et résultats

Figure A.1 présente le modèle de réseau considéré dans les expérimentations réalisées pour valider la première approche proposée. Dans la configuration expérimentale, 22 objets M5StickC [143] ont été déployés sur une surface de 300*300 m^2. Ces objets sont des dispositifs IoT basés sur microprocesseur ESP32 compatible Arduino pour son développement. Nous évaluons et comparons notre proposition d'algorithme de routage appliqué avec MOGWO par rapport à un algorithme multi-objectif récent, le NSGA-III [59].

Tableau A.1 présente les résultats en termes de l'indicateur Hypervolume obtenus par les deux algorithmes de routage comparés. L'Hypervolume permet d'évaluer la convergence et la distribution des solutions générées par les deux algorithmes d'optimisation comparés. Trois scénarios de tests ont été exécutés, variant le nombre de générations et le nombre d'objectifs considérés. En conclu-

Figure A.1: Architecture réseau proposée



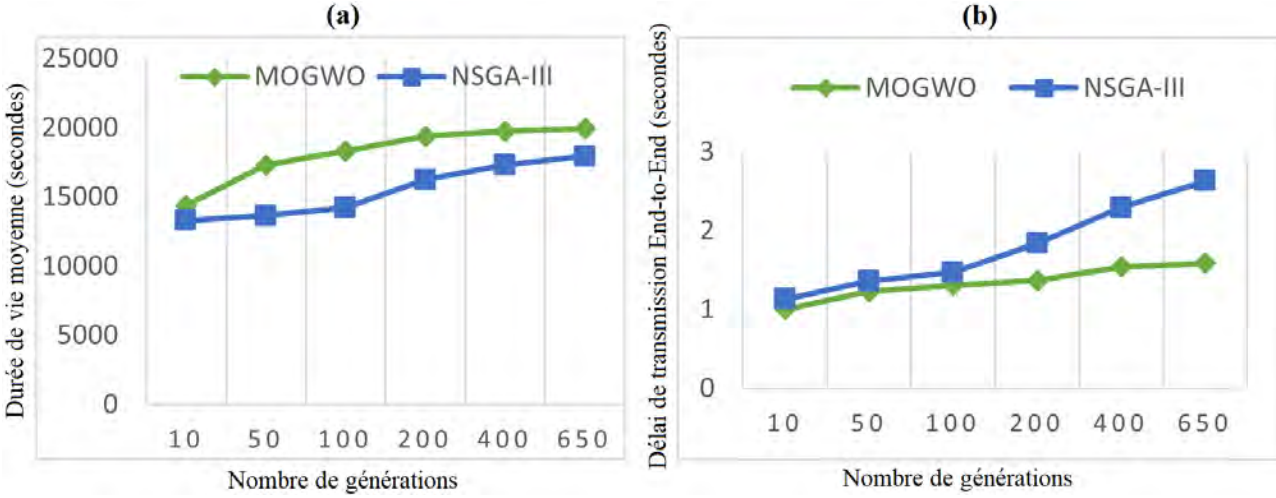
sion, les résultats indiquent que MOGWO présente généralement de meilleures performances en termes d'Hypervolume par rapport à NSGA-III dans différents scénarios de tests.

Table A.1: Les valeurs de l'indicateur Hypervolume pour MOGWO et NSGA-III appliqués au problème de routage multi-objectifs

Nombre de générations	Nombre d'objectifs	Valeur	MOGWO	NSGA-III
350	1	Meilleure	**0.921594**	0.921185
		Moyenne	0.914165	**0.916874**
		Pire	**0.901168**	0.881268
450	2	Meilleure	**0.982056**	0.981134
		Moyenne	0.980005	**0.981023**
		Pire	**0.979087**	0.978096
650	3	Meilleure	**0.973267**	0.921185
		Moyenne	**0.972601**	0.916874
		Pire	**0.972503**	0.972232

Figure A.2(a) compare les deux algorithmes de routage en termes de la durée de vie moyenne du réseau obtenus pour les différents nombres de générations exécutées. Ce critère exprime la capacité des solutions de routage à offrir une optimisation efficace de la consommation énergétique et à maintenir un équilibre de charge entre les objets du réseau. Globalement, nous observons une augmentation de la durée de vie moyenne du réseau avec le nombre croissant de générations pour les deux algorithmes. Cependant, les résultats indiquent que MOGWO offre une meilleure capacité à optimiser la consommation énergétique et à maintenir la durée de vie moyenne du réseau par rapport à NSGA-III.
Figure A.2(b) présente les résulats d'évaluation de deux algorithmes comparés en termes de délai de transmission entre la source initiale et la destination finale. L'analyse des résultats montre que, en moyenne, le délai de transmission est plus court lorsque le routage est effectué avec MOGWO

Figure A.2: La durée de vie moyenne du réseau et le délai de transmission end-to-end pour le routage modélisé avec MOGWO et NSGA-III



par rapport à NSGA-III pour chaque nombre de générations considéré. Cette capacité à offrir des délais de transmission plus courts montre une meilleure efficacité dans la transmission des données lorsque MOGWO est utilisé comme optimiseur multi-objectifs pour développer l'algorithme de routage proposé, ce qui peut contribuer à une meilleure réactivité du réseau et à une livraison plus rapide des données.

Les deux algorithmes sont comparés aussi en termes du nombre moyen de voisins des objets dans le réseau permettant d'évaluer la capacité des deux solutions de routage à maintenir une bonne répartition des objets dans l'environnement IoT étudié. De plus les valeurs RSSI moyennes sont comparées puisqu'elles décrivent la qualite des liens choisis. L'analyse comparative entre les deux algorithmes de routage multi-objectifs pour les réseaux IoT à topologie hybride, basés sur MOGWO et NSGA-III, révèle que MOGWO surpasse généralement NSGA-III sur plusieurs critères. MOGWO démontre une meilleure performance en termes d'Hypervolume, de durée de vie moyenne du réseau, de nombre de voisins des objets, de délai de transmission et de qualité du signal RSSI. Ces résultats indiquent que le routage proposé avec MOGWO offre une solution plus efficace et prometteuse pour le routage dans les réseaux IoT : une efficacité énergétique accrue, un équilibrage de charge amélioré, des délais de transmission réduits, des communications plus fiables et plus stables et le maintien d'une répartition efficace des objets dans le réseau.

Cette contribution et ses résultats font l'objet de notre publication [138].

A.3.4 Improved MOGWO

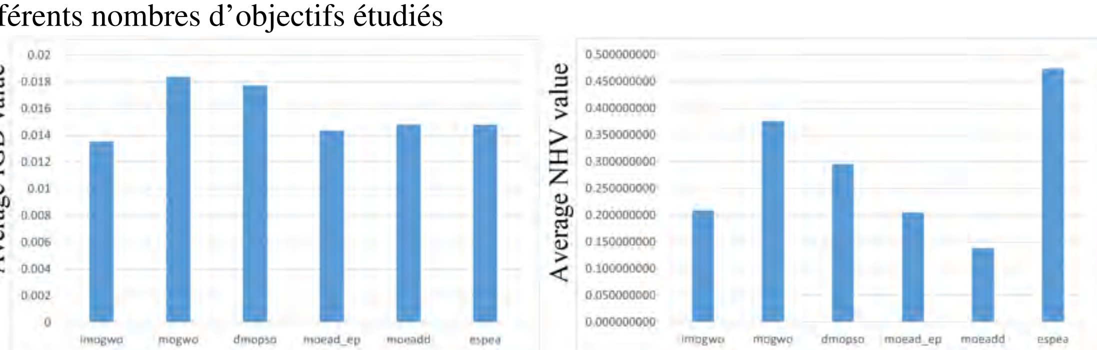
La revue de l'état de l'art a souligné l'importance de continuer les efforts dans la recherche pour créer des méthodes de résolution pour les problèmes d'optimisation en offrant des performances élevées sur des problèmes standard et des cas réels, tout en étant capables de traiter un plus grand nombre d'objectifs. Dans ce contexte, GWO [3] et MOGWO [2] sont deux variantes mono-objectif et multi-objectifs de la méta-heuristique GWO. Ils ont été développées et mises en oeuvre pour résoudre divers problèmes d'ingénierie. Toutefois, les performances de MOGWO restent limitées lorsqu'il s'agit de résoudre des problèmes comportant un grand nombre d'objectifs, puisqu'il est limité à résoudre des problèmes à seulement 2 ou 3 objectifs. Ainsi, notre troisième contribution consiste à proposer le IMOGWO, une version améliorée de MOGWO. IMOGWO ajuste les équations utilisées dans MOGWO afin d'améliorer sa capacité à résoudre efficacement des problèmes comportant un grand nombre d'objectifs, tout en garantissant une convergence efficace et une distribution optimale des solutions obtenues. Pour atteindre cet objectif, l'optimiseur proposé modifie la méthode d'exploration et les équations utilisées pour mettre à jour les positions des agents dans MOGWO.

Évaluation et résultats

L'objectif est d'utiliser IMOGWO pour élaborer un routage géographique multi-objectifs performant pour les réseaux IoT. Cependant, pour une évaluation préliminaire, nous avons testé ses performances en résolvant les problèmes de référence DTLZ [4]. Ces derniers sont parmi les problèmes multi-objectifs de référence les plus difficiles et populaires. Ils sont évolutifs car ils permettent de faire varier le nombre de variables et d'objectifs. Nous avons mené une série de tests sur ces problèmes en variant le nombre d'objectifs et le problème étudié. L'algorithme proposé (IMOGWO) a été comparé à MOGWO ainsi qu'à d'autres algorithmes d'optimisation multi-objectifs récents et bien connus : MOEA/D-IEps [69], MOEA/DD [66], ESPEA [64] et DMOPSO [63]. Les algorithmes sont comparés en termes de deux indicateurs : le IGD et le NHV. Nous avons sélectionné ces deux indicateurs pour leur large utilisation en tant que critères d'évaluation, fournissant une vision complète des performances de convergence et de distribution de Pareto Front [76].

Figure A.3 compare les valeurs moyennes des indicateurs IGD et NHV obtenus par les algorithmes d'optimisation testés pour résoudre tous les problèmes DTLZ1-4,7 avec différents nombres d'objectifs considérés. L'indicateur IGD offre une vue complète des performances de convergence et de distribution d'un optimiseur multi-objectif. IMOGWO se distingue avec la plus faible valeur moyenne d'IGD, montrant sa capacité à fournir des résultats compétitifs et prometteurs. MOEA/D-IEps, MOEA/DD et ESPEA présentent des valeurs proches, surpassant DMOPSO et MOGWO, qui présentent les pires performances moyennes d'IGD. En revanche, MOEA/DD offre la meilleure performance moyenne générale de NHV, suivi par IMOGWO qui occupe la troisième place. DMOPSO et MOGWO sont classés quatrième et cinquième, respectivement. Enfin, ESPEA présente la valeur moyenne globale la plus élevée de NHV.

Figure A.3: Valeurs moyennes des indicateurs IGD et NHV sur tous les problèmes DTLZ1-4,7 et les différents nombres d'objectifs étudiés



De plus, Tableau A.2 récapitule le nombre de fois où chaque algorithme a fourni les meilleures valeurs moyennes d'IGD et de NHV, sachant que le total des cas de test effectués (en variant le problème et le nombre d'objectifs) est de 25 (5 problèmes différents * 5 nombres d'objectifs différents). IMOGWO a ainsi enregistré les meilleurs résultats en termes d'IGD et de NHV dans respectivement 15 et 11 cas sur 25.

Table A.2: Le nombre de fois où chaque algorithme a fourni les meilleures valeurs moyennes des indicateurs IGD et NHV

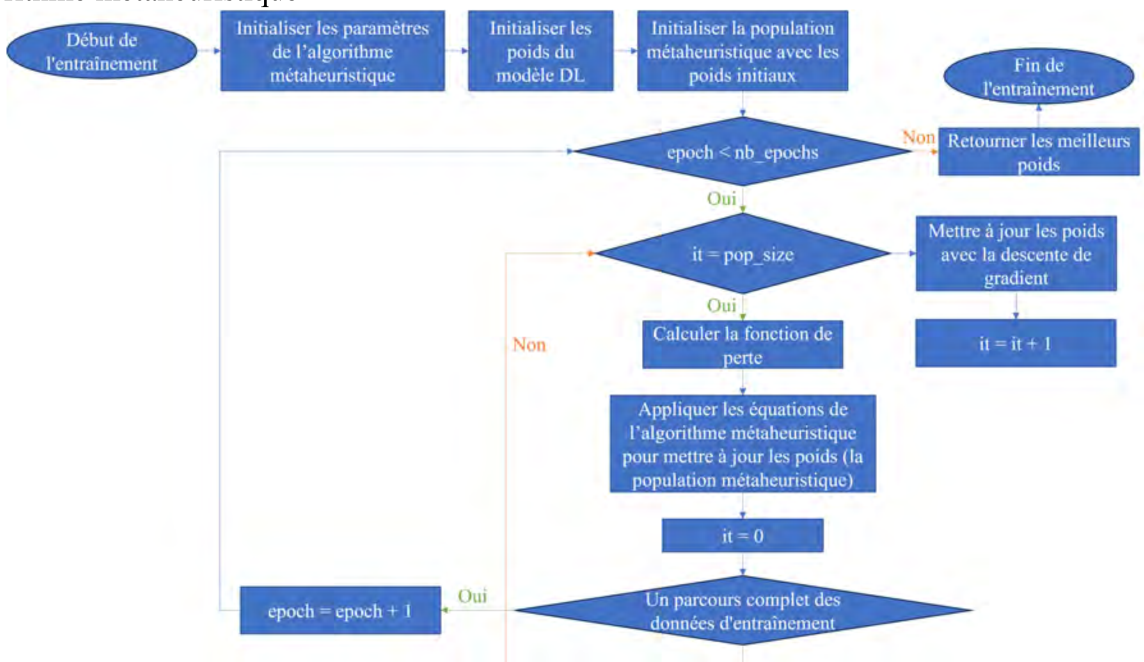
Algorithme		IMOGWO	MOGWO	DMOPSO	MOEA/D-IEps	MOEA/DD	ESPEA
Classé comme premier pour	IGD	15	0	0	3	3	4
	NHV	11	0	0	1	7	6

Les résultats des tests effectués démontrent la capacité de IMOGWO à atteindre une bonne convergence et une distribution efficace. Les solutions obtenues lors de la résolution des problèmes multi-objectifs testés convergent de manière satisfaisante vers les véritables Pareto fronts. Cette convergence élevée peut s'expliquer par les ajustements apportés par IMOGWO aux équations de changement de coordonnées des agents de recherche, améliorant ainsi leur proximité avec la proie modélisée par MOGWO. De plus, l'utilisation des poids pour classer les meilleures solutions a permis d'améliorer la convergence vers les solutions finales produites par IMOGWO. Cette capacité de convergence est également justifiée par la mise à jour des coordonnées des agents de recherche dès que de nouvelles positions améliorent les valeurs des objectifs. En outre, l'amélioration de la mise à jour des coordonnées des agents de recherche par la stratégie "dimension learning-based hunting" a renforcé l'exploration et la distribution des solutions obtenues par IMOGWO.
En plus des comparaisons standards, des tests statistiques inférentiels ont été utilisés pour comparer les résultats IGD et NHV de IMOGWO à ceux des autres optimiseurs étudiés. Le test de Friedman avec l'extension Iman-Davenport et le test de Holm [149] appliqués ont confirmé que les performances de IMOGWO en termes d'IGD sont significativement meilleures que celles de MOGWO, DMOPSO, MOEA/D-IEps et ESPEA. Bien que les résultats IGD de IMOGWO soient légèrement inférieurs à ceux de MOEA/DD, les tests statistiques ont montré que la différence n'est pas significative. De même, en ce qui concerne l'indicateur NHV, IMOGWO surpasse significativement MOGWO et ESPEA, tandis que les différences avec DMOPSO et MOEA/D-IEps sont moins significatives. Par ailleurs, les performances de MOEA/DD en termes de NHV se révèlent être les meilleures.

A.3.5 Optimisation des paramètres DL avec métaheuristiques

La revue de l'état de l'art a mis en évidence l'importance de considérer l'optimisation des poids des modèles Deep Learning à l'aide de métaheuristiques. Notre quatrième contribution, s'inscrit dans ce contexte. Nous proposons une approche de création d'un optimiseur conçu spécifiquement pour l'entraînement de modèles de DL en se basant sur une métaheuristique. Cette approche consiste à modéliser l'entraînement d'un modèle DL sous forme d'un problème d'optimisation mono-objectif. Pour ce faire, nous établissons une correspondance entre l'ajustement des paramètres des couches de neurones et la résolution d'un problème d'optimisation à l'aide d'un algorithme métaheuristique. L'originalité de notre proposition réside dans le fait que l'optimiseur proposé est applicable à plusieurs techniques de DL, plutôt que spécifique à une seule, ce qui le rend adaptable à différentes architectures de réseaux de neurones. L'objectif principal de cette approche est d'accélérer la convergence des poids des modèles DL vers des valeurs optimales tout en améliorant leurs performances.

Figure A.4: Organigramme de l'entraînement des poids des couches de neurones en appliquant un algorithme métaheuristique



La représentation de l'optimisation des poids à l'aide d'une méthode métaheuristique est illustrée dans Figure A.4.
Au début de la phase d'entraînement, les poids sont initialisés et un nombre maximal d'epochs est défini. Ces poids initiaux servent à démarrer la population de l'algorithme d'optimisation métaheuristique. Ensuite, une nouvelle itération est lancée jusqu'à ce que le nombre maximal d'epochs soit atteint. À chaque epoch, l'ensemble des données d'entraînement est utilisé, divisé en lots pour le traitement. À la fin de chaque lot, la valeur de la fonction de perte est calculée. Il est ensuite vérifié si la population de l'optimisation métaheuristique avait été entièrement construite ou mise à jour avec les dernières itérations. Dans ce cas, les équations mathématiques de l'optimisation métaheuristique sont appliquées pour calculer de nouveaux poids, et la population de l'algorithme est mise à jour avec ces nouveaux poids. Si ce n'est pas le cas, les poids sont mis à jour en utilisant les équations de la descente de gradient. Une fois que le nombre maximal d'epochs était dépassé, les poids obtenus lors de la dernière itération sont considérés comme optimaux, marquant ainsi la fin de la phase d'entraînement.
Notre approche de localisation en intérieur utilise cette méthode d'optimisation proposée visant à prédire avec précision les positions des objets dans le réseau IoT.

A.3.6 Localisation en intérieur en hybridant des techniques DL et des méta-heuristiques

Dans une cinquième approche, nous proposons alors une approche de localisation en intérieur dans un réseau IoT. Elle repose sur notre précédente contribution, puisqu'elle consiste à développer un modèle de Deep Learning optimisé et entraîné à l'aide de l'algorithme d'optimisation métaheuristique GWO [3]. Le processus se décompose en deux étapes distinctes. Tout d'abord, dans la phase hors ligne, un modèle de localisation est entraîné à l'aide de données préparées à l'avance, principalement des mesures de temps de vol UWB entre des objets connectés et des ancres fixes. Un algorithme d'optimisation métaheuristique guide l'entraînement du modèle de Deep Learning pour améliorer sa précision. Ensuite, dans la phase en ligne, les objets mobiles interagissent avec les ancres fixes pour obtenir des mesures de temps de vol UWB. Ces données sont ensuite transmises au modèle de localisation préalablement entraîné, qui estime la position des objets mobiles.

Expérimentations et résultats

Nous avons évalué les performances de notre solution en la comparant à trois autres méthodes de localisation largement utilisées : une méthode basée sur le Deep Learning [154], le IWCL [155], et une méthode de multilatération [156].

Le dataset utilisé pour les expérimentations est celui proposé par la plateforme réelle LocURa4IoT [140]. Ce dataset est spécialement conçu pour élaborer et évaluer des méthodes de localisation en intérieur dans les environnements IoT, en se basant sur les mesures de Time-of-Flight (ToF) et la technologie UWB. Ce dataset est disponible en ligne [141].

Le MAE et le MSE sont parmi les critères d'évaluation que nous avons utilisés pour comparer les solutions de localisation. Figure A.5 présente une étude comparative des valeurs MAE et MSE calculées à partir des prédictions obtenues par les solutions comparées. Ces résultats montrent que l'approche développée a fourni des erreurs MAE et MSE plus faibles pour les trois mesures considérées (les coordonnées x et y et la position). En fait, le taux auquel la solution de localisation proposée a atteint la distance minimale entre les positions réelles et estimées est de 93.54%. En outre, Tableau 5.18 offre une analyse exhaustive de différentes solutions de localisation, évaluées selon plusieurs critères de performance clés. Notre solution proposée a surpassé les autres méthodes en atteignant une précision de localisation de 98.92%, tandis que la méthode de multilatération est en deuxième rang avec une précision de 89.24% (avec un seuil de 0.5 mètre). Parmi les solutions comparées dans le tableau A.3, l'approche proposée se distingue comme étant la plus précise et fiable. Elle atteint une précision de localisation remarquable de 98.92% avec une erreur de position moyenne de seulement 0.057 mètre. De plus, elle présente un score R2 de 0.998, indiquant une prédictibilité élevée des positions. Cette méthode excelle également en termes de rappel (0.978) et de score F1 (0.996), démontrant sa capacité à identifier correctement les positions dans la grande majorité des cas. Cependant, elle nécessite un temps d'entraînement plus long de 328.302 secondes, bien qu'elle offre une estimation rapide en seulement 0.155 seconde. Bien que la solution développée ait surpassé les autres méthodes en termes de temps d'estimation, il est important de noter que la solution proposée nécessitait une phase préliminaire hors ligne, demandant 328.302 secondes pour entraîner le modèle de prédiction. D'autre part, la méthode de multilatération et le IWCL ne nécessitent aucune étape supplémentaire en dehors de l'estimation de position en temps réel.

Pour démontrer la supériorité de notre proposition, nous avons appliqué aussi des tests statistiques inférentiels, notamment le test de Friedman avec l'extension d'Iman-Davenport et la procédure post-hoc de Holm. Les résultats de ces comparaisons ont révélé que notre approche de localisation proposée parvient à estimer les positions de manière cohérente et précise, surpassant ainsi les autres solutions de localisation testées.

Plusieurs conclusions ressortent des tests réalisés et des résultats obtenus :

- En utilisant l'optimisation basée sur GWO pour entraîner les paramètres du modèle DL, tel que proposé dans cette approche, nous constatons des performances supérieures par rapport aux deux autres méthodes d'optimisation (AOA et GD). L'approche GWO accélère la convergence vers les paramètres optimaux et améliore le processus d'apprentissage du modèle DL.

- Les positions prédites à l'aide des mesures UWB ToF sont plus précises que celles obtenues avec le RSSI et le Range.
- La solution de localisation proposée présente une erreur moyenne de seulement 0.057 mètres entre les positions réelles et prédites, avec une précision de localisation de 98.92%. Elle parvient à estimer les positions de manière cohérente et précise, surpassant ainsi les autres

Figure A.5: Comparaison des valeurs moyennes globales des erreurs MAE et MSE pour les prédictions des coordonnées x et y ainsi que des positions (distances entre les positions réelles et estimées) pour les solutions comparées

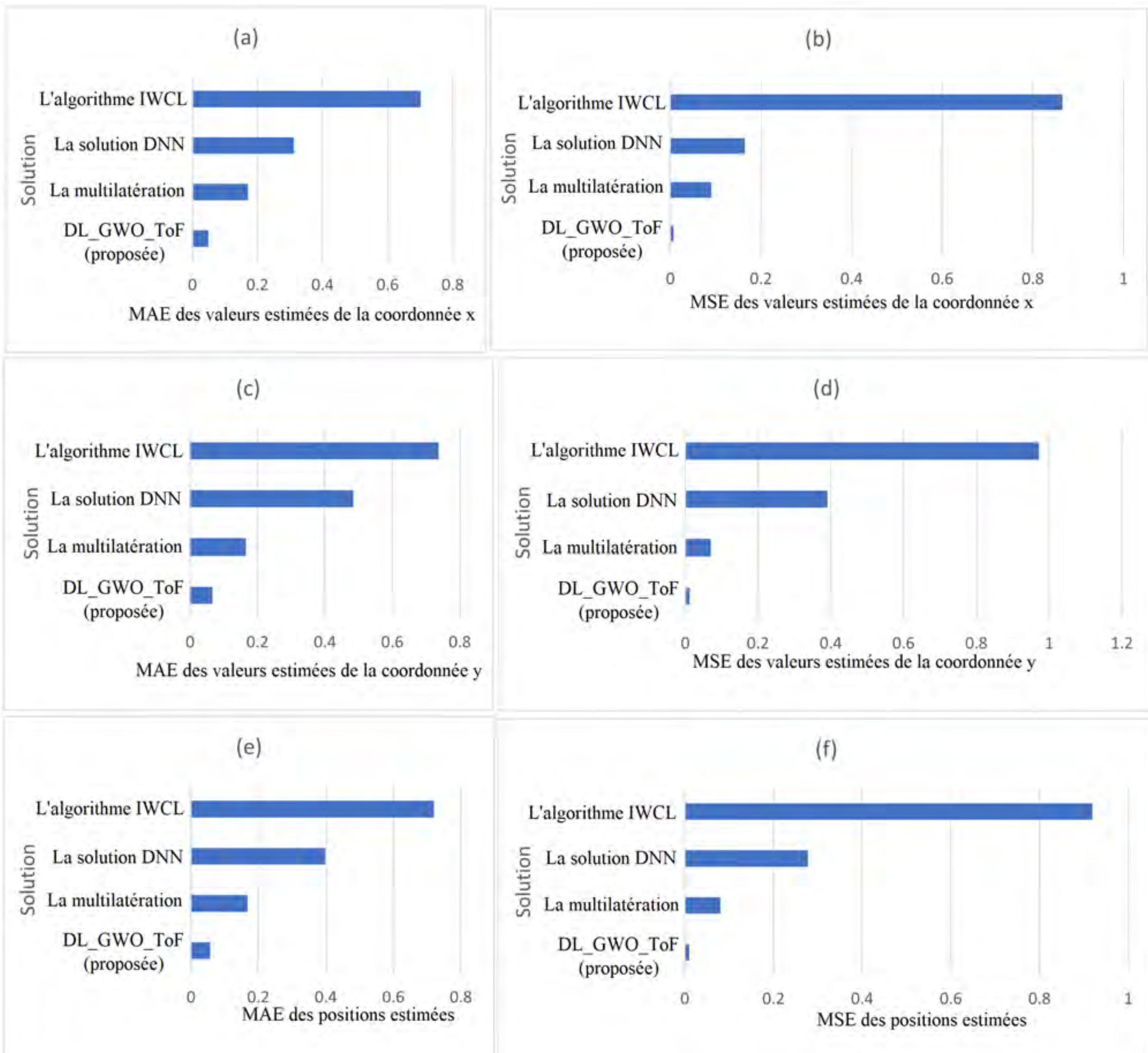


Table A.3: Comparaison de la solution proposée et d'autres méthodes de localisation existantes

Solution	Précision de localisation	Erreur de position moyenne (mètre)	R2	Recall	F1-score	Temps d'entraînement (sec)	Temps d'estimation (sec)
DL-GWO-ToF (proposée)	98.92%	0.057	0.998	0.978	0.996	328.302	0.155
Multilatération	89.24%	0.167	0.983	0.805	0.969	-	1.982
Solution DNN	51.61%	0.397	0.942	0.377	0.829	335.13	0.177
Algorithme IWCL	19.35%	0.719	0.817	0.120	0.562	-	0.064

solutions de localisation testées (notamment celles basées sur le DNN, l'algorithme IWCL et la méthode de multilatération).

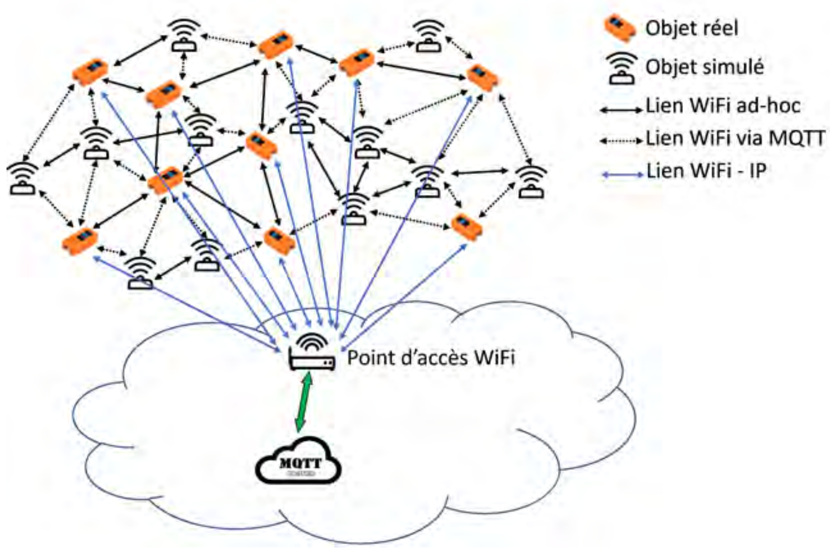
- Cette approche proposée surpasse les solutions de localisation en intérieur récentes, dont les meilleurs résultats atteignent une erreur moyenne de quelques dizaines de centimètres.
- Bien que l'approche proposée offre une estimation rapide par rapport aux autres solutions comparées, elle nécessite un temps d'entraînement plus long. La réduction du temps d'entraînement peut être une perspective envisageable pour améliorer cette approche. De plus, le réentraînement du modèle de localisation est nécessaire en cas d'évolutions ou de changements dans la topologie du réseau (nombre et positions des ancres). Par conséquent, le développement d'un modèle de DL capable de s'auto-apprendre et de s'auto-adapter à ces changements s'avère être une autre piste à explorer.

Cette contribution et ses résultats font l'objet de notre publication [142].

A.3.7 Un routage géographique hybride et multi-objectifs basé sur IMOGWO

Nous proposons enfin une approche de routage géographique hybride et multi-objectifs destinée à être appliquée dans les réseaux IoT à architecture maillée. Le routage géographique nécessite une connaissance des positions des objets connectés pour diriger les messages à travers le réseau. Pour cela, notre méthode exploite l'approche de localisation préalablement décrite, garantissant ainsi une connaissance précise des positions des objets. De plus, notre solution de routage est qualifiée de multi-objectifs, car elle applique notre proposition IMOGWO pour optimiser le processus de routage en tenant compte de divers objectifs simultanément. Ces objectifs incluent la progression positive et efficace vers la destination finale, une gestion optimale de l'énergie, une fiabilité globale dans les communications, la continuité des sauts jusqu'à destination finale, ainsi que l'optimisation de la latence de transmission. L'objectif global est d'assurer un acheminement efficace des messages tout en répondant aux exigences spécifiques des réseaux IoT, en établissant des routes multi-sauts optimisées. Le routage hybride implique la conservation de la connaissance locale de la topologie au niveau de chaque objet, jusqu'à une distance de 2 sauts, par le biais d'échanges périodiques de messages de contrôle. Ainsi, chaque objet construit une table de voisinage contenant des informations sur ses voisins accessibles à 1 et 2 sauts, actualisée par les messages de contrôle reçus. Par conséquent, les routes vers les voisins à 1 et 2 sauts sont connues de manière proactive, tandis que le chemin complet, de bout en bout, est découvert de manière réactive. Cette approche de routage s'intégre dans le domaine de l'Edge Computing. En effet, la construction des routes est

Figure A.6: Architecture réseau proposée



distribuée entre les objets du réseau IoT sans nécessiter d'ordinateur central pour gérer le routage, ce qui signifie une décentralisation du processus de prise de décision de routage. En d'autres termes, chaque objet où arrive le message en cours d'acheminement sélectionne la destination du prochain saut.

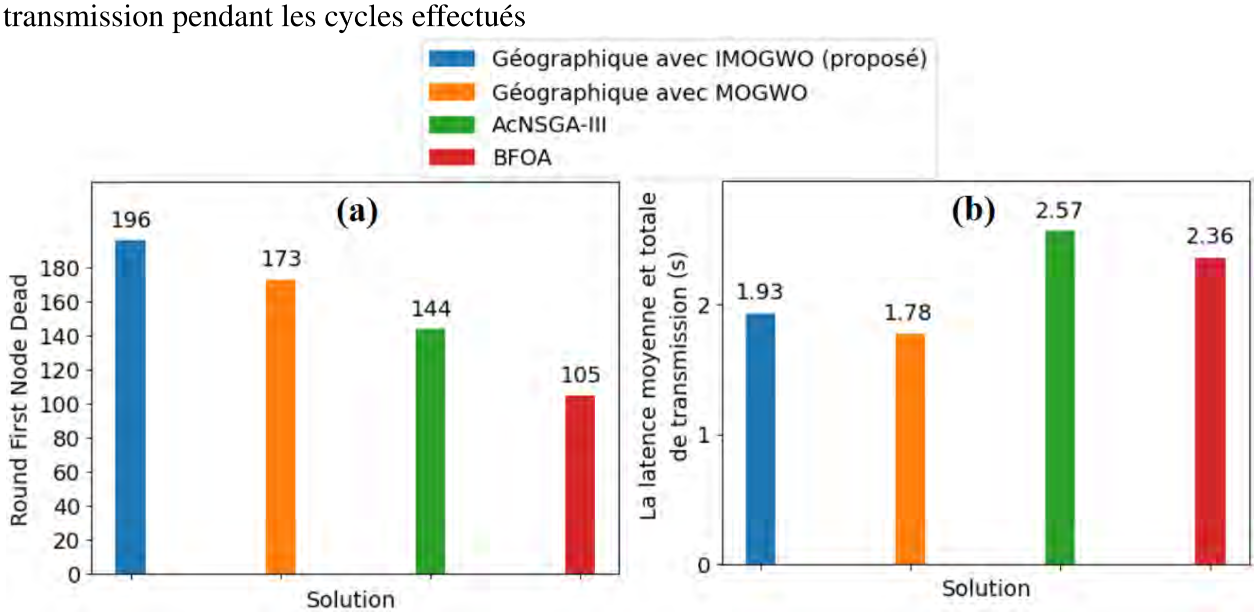
Expérimentations et résultats

Des simulations hybrides, combinant des éléments simulés et réels, ont été réalisées pour évaluer l'approche de routage géographique et hybride proposée. Nous avons choisi d'utiliser le simulateur IoT CupCarbon [157][158] pour réaliser les simulations hybrides. Les objets réels sont des appreils M5StickC [143]. Les objets réels et les objets simulés communiquent en utilisant le protocole MQTT [159]. Figure A.6 présente le modèle de réseau considéré dans les simulations réalisées pour valider cette approche proposée. Les performances de notre approche de routage développée avec IMOGWO ont été comparées à celles obtenues avec la même approche développée avec MOGWO, un algorithme de routage basé sur la QoS utilisant AcNSGA-III [83], et un algorithme de routage géographique basé sur BFOA [109]. Cette comparaison s'est basée sur divers indicateurs d'évaluation tels que : le FND, le PDR et la latence de transmission.

Figure A.7(a) compare les valeurs de FND pour les algorithmes de routage. Le FND désigne le moment où le premier objet épuise entièrement son énergie. En d'autres termes, cette figure compare les périodes de stabilité offertes par les solutions de routage étudiées. À partir de ces résultats, il apparaît clairement que le routage géographique avec IMOGWO et MOGWO démontre une performance supérieure par rapport à AcNSGA-III et BFOA, en assurant une meilleure gestion de la consommation énergétique et une meilleure stabilité dans la répartition de la charge entre les noeuds du réseau. En outre, l'utilisation de l'algorithme IMOGWO dans le routage géographique surpasse celle de MOGWO, en offrant une répartition plus équilibrée de la charge énergétique, ce qui se traduit par une capacité supérieure à gérer efficacement le problème de routage multi-objectifs.

Figure A.7(b) compare les valeurs de la latence moyenne de transmission pour l'ensemble des cycles effectués. La latence de transmission consiste au temps écoulé entre l'envoi d'un paquet de données par la source initialel et sa réception par la destination finale. Les résultats indiquent que notre proposition de routage géographique avec IMOGWO et MOGWO offre des performances supérieures en termes de latence de transmission par rapport aux algorithmes AcNSGA-III et BFOA. Entre les deux approches de routage géographique, celle utilisant MOGWO semble légèrement plus efficace en réduisant la latence moyenne totale. Cette différence pourrait être attribuée au fait que IMOGWO implique des calculs plus complexes que MOGWO, ce qui lui permet de mieux optimiser les chemins de transmission. Cependant, cela peut entraîner une légère augmentation des latences par rapport à MOGWO.
Figure A.8 présentent les valeurs de PDR total calculées pour l'ensemble des cycles effectués. Notre proposition de routage géographique avec IMOGWO offre le PDR total le plus élevé (92.2%), suivie par note proposition avec MOGWO (90.6%). En revanche, les algorithmes AcNSGA-III et BFOA présentent des performances inférieures, avec des PDR totaux de 86.6% et 85% respectivement. En conclusion, les résultats montrent que notre proposition du routage géographique appliquée avec IMOGWO surpasse les performances des autres algorithmes évalués en termes de fiabilité de la transmission des paquets dans le réseau. Cependant, une diminution du PDR à travers les cycles pour tous les algorithmes comparés peut être remarquée. Cette baisse peut s'expliquer par le fait que le nombre d'objets actifs diminue également au fil des cycles, réduisant ainsi le nombre d'objets disponibles pour effectuer les différents sauts pendant le routage.
En plus des comparaisons standards, des analyses par les tests statistiques inférentiels (le test de Friedman avec l'extension d'Iman-Davenport et la procédure post-hoc de Holm) ont été appliqués et ils ont confirmé la supériorité de notre approche de routage géographique et hybride par rapport aux autres algorithmes comparés. Cette efficacité consiste à une gestion optimisée de la consommation énergétique, une répartition équilibrée de la charge entre les noeuds du réseau, et des performances accrues en termes de latence de transmission et de taux de livraison des paquets.

Figure A.7: Comparaison des valeurs de FND et des valeurs de la latence moyenne et totale de transmission pendant les cycles effectués



Ces observations peuvent être justifiées par l'adaptabilité de l'approche de routage géographique aux changements dynamiques du réseau, ainsi que par la capacité des algorithmes d'optimisation appliqués, notamment IMOGWO, à trouver des chemins de transmission efficaces. En effet, les résultats ont montré que IMOGWO surpasse MOGWO dans la résolution du problème de routage géographique multi-objectifs. Par ailleurs, la diminution progressive du nombre d'objets actifs dans le réseau avec l'approche de routage géographique proposée reflète une meilleure gestion de l'énergie des objets, réduisant ainsi les risques d'épuisement d'énergie. Enfin, bien que présentant une légère augmentation en termes de données de contrôle, les approches de routage géographique présentent une meilleure fiabilité de la transmission des paquets, assurant ainsi une meilleure qualité de service dans l'ensemble du réseau.

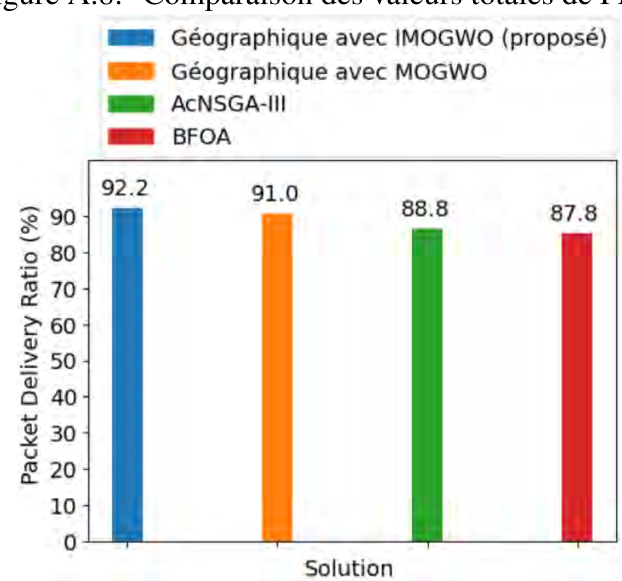
A.4 Conclusions et interprétations

Les principales contributions de nos travaux de recherche et leurs résultats peuvent être résumées comme suit :

- Nous avons mené une revue approfondie de la littérature en commençant par une introduction générale à IoT. Ensuite, nous avons étudié le routage et la localisation en intérieur dans les réseaux IoT. Par la suite, nous avons abordé les fondamentaux de l'optimisation et de l'apprentissage profond (DL). Dans l'état de l'art, des travaux récents liés à ces domaines ont également été présentés.

- Une modélisation mathématique du routage dans les réseaux IoT a été proposé. Il définit et formalise les caractéristiques du réseau IoT, les objectifs clés, les variables de décision et les contraintes liées au problème de routage. Ce modèle a été appliqué pour rendre les approches introduites plus structurées et simplifier leur évaluation. Il se concentre sur divers objectifs :

Figure A.8: Comparaison des valeurs totales de PDR



progression vers la destination finale, gestion efficace de l'énergie, connexions fiables entre les objets, continuité de la route et minimisation de la latence.

- Une approche de routage, basée sur l'algorithme MOGWO [2], a été développée pour traiter le routage comme un problème d'optimisation multi-objectifs. Notre objectif principal était de concevoir un algorithme de routage multi-sauts, réactif et multi-objectifs dans un réseau IoT à une topologie hybride. Les performances de cette approche de routage ont été évaluées à travers des expériences réelles. Les résultats ont été comparés à ceux de NSGA-III [59]. L'étude de ces résultats mettent en évidence que MOGWO offre une solution plus efficace et prometteuse pour le routage dans les réseaux IoT que NSGA-III. Cette approche garantit une meilleure efficacité énergétique, un meilleur équilibrage de charge, des délais de transmission réduits, des communications plus fiables et stables, ainsi qu'une distribution plus efficace des objets dans le réseau.

- Une version améliorée de la métaheuristique multi-objectifs MOGWO, appelée IMOGWO, a également été suggérée. Elle ajuste les équations fondamentales appliquées par MOGWO. Il vise à augmenter l'efficacité de MOGWO dans la résolution de problèmes complexes avec plusieurs objectifs, en assurant une convergence efficace et une distribution optimale des solutions obtenues. Notre objectif est de développer un routage géographique multi-objectif dans les réseaux IoT en se basant sur IMOGWO. Dans le cadre d'une évaluation préliminaire, les performances de IMOGWO ont été testées en résolvant les problèmes de référence DTLZ [4]. Les résultats fournis par IMOGWO ont été comparés à ceux obtenus par les optimiseurs suivants : MOGWO, DMOPSO [63], MOEA/D-IEps [69], MOEA/DD [66] et ESPEA [64]. Les indicateurs d'évaluations appliqués sont IGD et NHV. Des comparaisons standard et des tests statistiques inférentiels ont été réalisés pour montrer que IMOGWO présente une convergence plus efficace et une distribution optimale des solutions obtenues, par rapport aux autres optimiseurs.

- Une nouvelle technique a été introduite pour optimiser l'entraînement des modèles de DL en utilisant une métaheuristique mono-objectif. Son originalité réside dans le fait que l'optimiseur développé peut être appliqué à plusieurs techniques de DL, ce qui le rend adaptable à différentes architectures de réseaux de neurones. La méthode a ensuite été utilisée pour développer l'approche de localisation intérieure dans les réseaux IoT.

- Nous avons proposé une approche de localisation en intérieur pour les réseaux IoT, basée sur notre approche précédemment suggérée. Elle entraîne un modèle DL avec des mesures ToF prises par la technologie UWB en appliquant l'algorithme d'optimisation métaheuristique GWO [3]. Les performances de notre solution ont été comparées à celles de trois autres techniques de localisation largement utilisées : une méthode basée sur le DL [154], le IWCL [155] et une technique de multilatération [156]. Pour démontrer la supériorité de l'approche introduite, des comparaisons standard et des tests statistiques inférentiels ont été réalisés. Les résultats de la comparaison ont révélé que l'approche de localisation développée assure une estimation meilleure et plus précise des positions et surpasse les autres solutions de localisation testées. En effet, la méthode suggérée a fourni une erreur moyenne de seulement 0.057 mètres entre les positions réelles et prédites et une précision de localisation de 98.92%.

Ces résultats sont meilleurs que ceux fournis par d'autres solutions récentes de localisation intérieure qui ont donné une erreur moyenne d'une dizaines de centimètres.

- Nous avons développé une approche de routage géographique hybride et multi-objectif conçue pour les réseaux IoT ayant une architecture maillée. La nature géographique de l'algorithme proposé nécessite de déterminer les positions des objets dans le réseau, ce qui est fourni par notre approche de localisation en intérieur proposée. Le processus de sélection de route multi-sauts a été optimisé selon divers objectifs simultanément en utilisant IMOGWO proposé. Des simulations hybrides, combinant des objets simulés et des objets réels, ont été réalisées pour évaluer l'approche de routage géographique et hybride proposée. Les performances de notre approche de routage reposant sur IMOGWO ont été comparées à celles de la même approche basée sur MOGWO, un algorithme de routage basé sur la qualité de service utilisant AcNSGA-III [83] et un algorithme de routage géographique reposant sur le BFOA [109]. Cette comparaison a été réalisée en utilisant divers critères d'évaluation tels que l'efficacité, la fiabilité, l'optimisation énergétique, l'équilibrage de charge et la stabilité. Des comparaisons standard et des analyses par des tests statistiques inférentiels ont montré que IMOGWO surpasse MOGWO dans la résolution du problème de routage géographique multi-objectifs. Ils ont également confirmé la supériorité de notre approche de routage géographique et hybride par rapport aux autres algorithmes comparés. L'approche proposée a montré une efficacité dans la gestion de la consommation d'énergie, l'équilibrage de charge, la latence de transmission et le taux de livraison des paquets. Malgré une légère augmentation des données de contrôle, les approches de routage géographique, utilisant MOGWO et IMOGWO, fournissent une meilleure fiabilité de transmission et garantissent, par conséquent, une meilleure qualité de service dans tout le réseau.

En résumé, les évaluations des approches suggérées à travers des expérimentations réelles et des simulations hybrides, l'utilisation des indicateurs de performance adaptés, leurs comparaisons avec des solutions existantes et les analyses statistiques réalisées mettent en évidence le bon comportement et l'efficacité de nos approches sur les défis de routage et de localisation en intérieur dans les réseaux IoT.

A.5 Perspectives et directions de recherche futures

Afin d'explorer en profondeur les futures directions de recherche et les implications pratiques de cette thèse, nous abordons dans cette section les perspectives que la thèse ouvre dans le domaine.

- Le déploiement des approches proposées, de localisation et de routage, et leur implémentation dans la plateforme LocURa4IoT [140] en tenant compte des contraintes matérielles et des exigences de performance.

- Approfondissement des aspects théoriques : approfondir la modélisation mathématique et théorique des problèmes de routage et de localisation en milieu intérieur dans les réseaux IoT, en explorant de nouveaux cadres théoriques.

- Optimisation de l'algorithme IMOGWO : une perspective pour réduire le temps d'exécution et la complexité des calculs dans notre métaheuristique IMOGWO serait d'explorer des modifications supplémentaires dans les processus de mise à jour des positions des loups. Par exemple, nous pourrions envisager une analyse approfondie des phases de recherche et d'exploration de l'algorithme pour révéler des opportunités pour optimiser la sélection des loups leaders et la gestion des solutions non dominées. En intégrant ces ajustements, nous visons à élaborer une version améliorée d'IMOGWO qui maintient son efficacité tout en réduisant le temps d'exécution et la complexité des calculs, offrant ainsi une solution plus compétitive pour la résolution des problèmes multi-objectifs.

- Auto-apprentissage et auto-adaptation du modèle de Deep Learning pour la localisation : nous visons à concevoir un système de localisation capable de détecter et de réagir aux changements dans l'environnement, tels que l'ajout ou le déplacement des ancres, ainsi que la présence d'obstacles temporaires. L'objectif est de développer un modèle de Deep Learning capable de s'auto-apprendre et de s'auto-adapter aux changements dans l'environnement et les conditions du réseau. En d'autres termes, ce modèle ne nécessite pas d'être ré-entrainé totalement de nouveau lorsqu'il y a des changements dans l'environnement. Il évolurait en continu au fur et à mesure qu'il serait exposé à de nouvelles données, garantissant ainsi une localisation en temps réel et une adaptation dynamique aux nouveaux scénarios. Cette adaptabilité permettrait au système de maintenir des performances de localisation précises et fiables malgré les évolutions de la topologie.

- Fusion de données multi-modales : investiguer des méthodes pour fusionner efficacement les données provenant de différentes sources sensorielles (par exemple, UWB, Wi-Fi, Bluetooth, etc.) à l'aide de techniques de Deep Learning, afin d'améliorer la précision de la localisation en milieu intérieur dans les réseaux IoT.

- Routage basé sur l'Intelligence Artificielle : intégrer des techniques d'Intelligence Artificielle telle que l'apprentissage par renforcement dans les algorithmes de routage pour améliorer l'efficacité et l'adaptabilité du routage dans les réseaux IoT. L'apprentissage par renforcement est une branche de l'apprentissage automatique où un agent apprend à prendre des décisions en interagissant avec un environnement. L'objectif de l'agent est de maximiser une récompense cumulative à long terme en choisissant des actions appropriées dans différentes situations.

- Analyse Big Data pour l'optimisation du routage et de localisation en intérieur : explorer l'utilisation des techniques Big Data de traitement de flux de données en temps réel, telles que Apache Kafka [160] ou Apache Flink [161], pour l'analyse en temps réel et continue des données générées par les réseaux IoT. Ces données peuvent inclure celles de localisation, de routage, de trafic et de qualité de service.

- Edge Computing : explorer les opportunités offertes par le Edge Computing pour améliorer les performances du routage et de la localisation en intérieur dans les réseaux IoT. Le Edge Computing implique le traitement des données et la prise de décision au plus près de leur origine (les capteurs ou les noeuds de traitement à proximité). Contrairement au modèle traditionnel où les données sont envoyées à un point central ou au cloud pour être traitées, le Edge Computing déplace une partie de ce traitement vers les extrémités du réseau, réduisant ainsi la latence et la dépendance de la centralisation et au cloud.

- Applications spécifiques : explorer des applications spécifiques pour les solutions de routage et de localisation en milieu intérieur dans des domaines tels que la santé, la logistique, les villes intelligentes et les bâtiments intelligents, en identifiant les cas d'utilisation pertinents et en adaptant les algorithmes pour prendre en compte les caractéristiques et les contraintes propres à chaque contexte.

- Optimisation multi-objectifs : approfondir la recherche sur les techniques d'optimisation multi-objectifs pour faire face aux défis de routage et de localisation en intérieur dans les réseaux IoT.

Titre : Routage et localisation en intérieur dans les réseaux IoT : approches intelligentes basées sur les métaheuristiques et l'apprentissage profond

Mots clés : Internet des objets, Optimisation, Apprentissage profond, Métaheuristiques, Routage dans les réseaux IoT, Localisation en intérieur

Résumé : L'expansion rapide de l'Internet des Objets (IoT) a révolutionné de nombreux aspects de notre vie quotidienne. Cependant, elle a également introduit de nouveaux défis, notamment en matière de localisation en intérieur puis d'optimisation de routage, en raison de la nature dynamique et de l'évolutivité topologique constante des réseaux IoT et de leur environnement de déploiement. En effet, un routage efficace et une localisation précise sont essentiels pour assurer le bon fonctionnement des applications IoT. Dans cette thèse, nous abordons ces défis par l'utilisation de l'IA, en combinant l'optimisation par métaheuristiques avec les techniques d'apprentissage profond. Notre objectif principal est donc de proposer une approche hybride de localisation en intérieur précise d'objets connectés mobiles en combinant l'apprentissage profond avec les métaheuristiques. Puis, nous réinvestissons cette méthode pour développer un routage géographique et multi-objectifs pour les réseaux IoT à architecture maillée. Cette approche de routage est qualifiée de multi-objectifs, car elle utilise l'optimisation métaheuristique pour prendre en compte plusieurs critères simultanément. Son objectif est d'améliorer les performances du réseau en réduisant les retards de transmission et en maximisant l'utilisation des ressources disponibles, assurant ainsi des performances optimales pour les applications IoT. Nos contributions sont évaluées à travers des expérimentations réelles, des simulations et des études comparatives démontrant que les approches introduites surpassent d'autres solutions existantes, en se basant sur plusieurs métriques de comparaison combinées.

Title: Routing and indoor localization in IoT networks: intelligent approaches based on metaheuristics and deep learning

Key words: Internet of Things, Optimization, Deep Learning, Metaheuristics, Routing in IoT Networks, Indoor localization

Abstract: The rapid expansion of the Internet of Things (IoT) has revolutionized many aspects of our daily lives. On the other hand, it has led to the appearance of new challenges, particularly in indoor localization and routing optimization, due to the dynamic nature and constant topological scalability of IoT networks and their deployment environment.

In fact, efficient routing and accurate indoor location ensure the adequate functioning of IoT applications. In this thesis, we deal with these challenges through the use of AI by combining metaheuristic optimization with Deep Learning techniques. The main objective of this work is to develop a hybrid approach for precise indoor localization of mobile connected objects by combining deep learning with metaheuristics. Then, the proposed method is used to develop geographic and multi-objective routing in IoT networks having mesh architecture. This routing approach is called multi-objective as it utilizes metaheuristic optimization to consider simultaneously several criteria. It is essentially applied to improve the network performance, by minimizing the transmission delays and maximizing the employment of the available resources, and, thereby, to ensure the optimal performance of IoT applications. Our contributions are evaluated through real experiments, simulations and comparative studies demonstrating that the introduced approaches outperform other existing solutions, based on several comparison metrics combined.