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

Remerciements

Je souhaite tout d'abord exprimer mon profond respect et ma gratitude à **Pr. Thierry VAL**, mon directeur de thèse, pour son soutien, sa disponibilité et sa compréhension. Je lui suis particulièrement reconnaissante de m'avoir accordé une grande liberté dans mes recherches, d'avoir examiné attentivement tous mes commentaires et idées, et d'avoir répondu à mes nombreuses questions. Ses conseils et sa guidance ont été précieux pour mener à bien ce travail. J'ai beaucoup apprécié son soutien et j'ai pris plaisir à nos échanges.

Je tiens à exprimer ma sincère gratitude à mon co-encadrant, **Dr. Sami MNASRI**, pour son intérêt pour ce travail. Je lui suis profondément reconnaissante pour ses conseils précieux et son encouragement. J'ai vraiment apprécié son soutien tout au long la durée du travail de thèse.

Je remercie également **Pr. Gérard CHALHOUB** et **Dr. Hanen IDOUDI** d'avoir accepté de faire partie du jury d'évaluation de mon travail.

Je tiens également à remercier **Pr. Adrien VAN DEN BOSSCHE** et **Dr. Cassandre VEY** d'avoir accepté d'examiner mon travail de thèse.

Je remercie également **Dr. Cassandre VEY** et **Dr. Guillaume GAILLARD** pour leur précieuse assistance.

Table des matières

Table des figures

Liste des Algorithmes

Liste des tableaux

Liste des acronymes

AcNSGA-III Ant-Colony Non-dominated Sorting Genetic Algorithm III
ACO Ant Colony Optimization
AENN Auto-encoder Neural Network
ANN Artificial Neural Network
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
CSO Chicken Swarm Optimization
D Diversity
DELTA Deep Learning-Based Cooperative Architecture
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
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
J Jitter
KF Kalman Filter
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
ML Machine Learning
MLE Maximum Likelihood Estimation
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
POM Problème d'optimisation multi-objectifs
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
ROI Large Region of Interest
RSSI Received Signal Strength Indicator
SA Simulated Annealing
SBX Simulated Binary Crossover
SDE Standard-Deviation of Energy
SFO Sun Flower Optimization
SGD Stochastic Gradient Descent
SNR Signal-to-Noise Ratio
SP Spacing Metric
SVR Support Vector Regression
TDoA Time Difference of Arrival

ToF Time of flight
TS Tabu Search
TWR Two-Way Ranging
UWB Ultra-Wide Band
UWL Uncertainty Weighted Localization
WFG Walking Fish Group
WFGHV Walking Fish Group Hypervolume
WSN Wireless Sensor Network

Chapitre 1

Introduction générale

Sommaire

1.1 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, révolutionnant 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 n'est pas sans défis. Parmi ceux-ci, le problème du routage dans les réseaux IoT maillés se pose comme un défi majeur à résoudre, notamment en raison de leur nature dynamique et de leur évolutivité topologique constante. Il consiste à déterminer les chemins optimaux 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, ce qui est crucial pour le bon fonctionnement des applications IoT. Tout d'abord, il permet d'optimiser les performances en réduisant les retards de transmission et en maximisant l'utilisation des ressources réseau, ce qui assure des performances optimales pour les applications IoT. 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 entre les appareils IoT. En outre, il contribue à une économie d'énergie significative, notamment dans les dispositifs alimentés par batterie, en limitant les transmissions

inutiles ou en privilégiant 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, ce qui garantit sa flexibilité et sa scalabilité. Il joue également un rôle crucial dans la sécurisation des données en appliquant des stratégies de routage sécurisées pour prévenir les attaques et protéger les informations sensibles transitant dans le réseau IoT. 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 Global Navigation Satellite Systems (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, résoudre efficacement les problèmes de routage et de localisation en intérieur dans les réseaux IoT devient impératif. Cependant, la complexité et la dynamique de ces réseaux exigent l'application de paradigmes et 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, permettant 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 Deep Learning (DL) ont démontré leur capacité à aborder de manière efficace les problèmes complexes, offrant des solutions innovantes et adaptatives. Grâce à leur aptitude à extraire des modèles à partir de données, le DL 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, présentées dans la section suivante.

1.2 Objectifs de recherche et principales contributions

Dans cette thèse, nous envisageons de tirer parti de ces deux paradigmes complémentaires, l'optimisation multi-objectifs et les techniques de DL, en les combinant pour proposer des solutions novatrices pour le routage et la localisation en intérieur dans les réseaux IoT.

— En première étape, nous avons effectué une revue approfondie de la littérature. L'état de l'art se concentre d'abord sur l'étude du routage et de la localisation en intérieur dans les réseaux IoT, en explorant les concepts clés nécessaires pour comprendre ces deux défis. Ensuite, il examine l'optimisation et le DL, explorant ainsi ces deux domaines innovants. De plus, il présente des études et des comparaisons des travaux récents liés aux thèmes abordés. Cet état de l'art vise à fournir une compréhension approfondie des principaux concepts et à constituer une base pour les contributions théoriques et expérimentales ultérieures de la thèse.

— 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. L'objectif est de fournir un cadre pour la conception et l'optimisation des algorithmes de routage dans les réseaux IoT, en mettant l'accent sur des objectifs essentiels tels que la réduction de la latence de transmission, l'amélioration du taux de livraison des paquets et la gestion efficace de l'énergie. Cette modélisation vise ainsi à établir une base pour le développement de solutions de routage robustes et efficaces.

— En troisième étape, une approche de routage est proposée. Elle consiste à l'application de l'algorithme MOGWO [2] afin de modéliser un routage réactif et multi-objectifs dans un réseau IoT à topologie hybride. Dans cette approche, le routage est abordé comme un problème d'optimisation dont l'objectif est de définir une route efficace à travers des sauts multiples. Cette approche intègre également deux objectifs essentiels : la gestion efficace de l'énergie et la réduction de la latence de transmission.

— L'objectif de la quatrième étape est de développer une procédure de routage géographique basée sur une solution de localisation en intérieur. À cette fin, les approches suivantes sont proposées :

  — Nous proposons une approche d'amélioration de l'algorithme métaheuristique multi-objectifs MOGWO pour répondre à ses limites de performances lors de la résolution de problèmes comportant un grand nombre d'objectifs. Cette amélioration, appelée Improved MOGWO (IMOGWO), modifie les équations fondamentales de MOGWO visant à avoir une exploration plus efficace et une mise à jour optimisée des positions des agents. Notre approche de routage géographique multi-objectifs s'appuie sur IMOGWO pour

sélectionner les routes multi-sauts lors de l'acheminement des données dans le réseau IoT.

— 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. L'objectif principal de cette approche est d'accélérer la convergence des modèles DL vers des valeurs optimales des poids tout en améliorant leurs performances.

— 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 DL optimisé et entraîné à l'aide de l'algorithme d'optimisation métaheuristique (Gray Wolf Optimizer (GWO)) [3]. Le processus se de localisation se repose principalement sur des mesures de temps de vol Ultra-Wide Band (UWB) entre des objets connectés et des ancres fixes pour entraîner le modèle DL et pour estimer les positions des objets mobiles.

— 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. 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. 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. Cette approche de routage s'intègre dans le domaine de l'Edge Computing, puisque la construction des routes est distribuée entre les objets du réseau IoT.

Ces approches proposées sont testées et évaluées en utilisant des indicateurs de performances adaptés. À cette fin, des expériences réelles et des simulations ont été menées. Les résultats obtenus sont comparés à ceux d'autres solutions existantes récentes en appliquant des comparaisons standards et des tests statistiques inférentiels pour démontrer les meilleures performances et la supériorité de nos propositions.

Dans ce manuscrit de thèse, nous proposons quatre chapitres pour présenter nos contributions. La manière dont le manuscrit est structuré sera décrite dans la section suivante.

1.3 Structuration du document

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, tandis que la seconde partie détaille les contributions proposées dans les chapitres 4 et 5.

Dans le chapitre 2, nous explorerons le routage et la localisation en intérieur dans le contexte des réseaux IoT. Nous commencerons 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 employées 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 DL. 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, 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 : l'optimisation des poids des modèles DL par des algorithmes métaheuristiques et le développement d'un modèle DL de localisation en intérieur basé sur les mesures UWB Time of flight (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 : évaluer les performances de notre solution de routage proposée, les comparer à celles d'autres solutions de routage existantes, et vérifier l'efficacité de l'algorithme IMOGWO dans un contexte de routage 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.

Première partie

État de l’art

Chapitre 2

Le routage et la localisation en intérieur dans les réseaux IoT

Sommaire

2.1 Introduction

Dans ce premier chapitre, nous aborderons le routage et la localisation en intérieur dans le contexte des réseaux IoT. Nous commencerons par une introduction générale à l'IoT en présentant ses topologies de réseau, ses technologies de communication, ses domaines d'application ainsi que ses défis majeurs. Ensuite, nous explorerons en détail deux aspects essentiels des réseaux IoT : le routage, en s'orientant particulièrement vers le routage géographique, et la localisation en intérieur. En premier lieu, nous nous intéressons au routage dans les réseaux IoT, en définissant ce concept et en explorant différents types de routage, ainsi que les indicateurs de performance pertinents. Nous finissons cette section par une revue des études récentes sur le routage dans un réseau IoT. Nous passons, en deuxième lieu, à la localisation en intérieur, en discutant de sa définition, de son importance et des différentes techniques utilisées, ainsi que des critères d'évaluation pertinents. Enfin, nous comparons et examinons des études récentes sur la localisation en intérieur dans les réseaux IoT, offrant ainsi un aperçu complet de ces deux domaines clés.

2.2 Introduction à l'IoT

L'Internet des Objets (IoT) représente une avancée technologique majeure qui révolutionne notre interaction avec le monde physique en connectant une multitude d'appareils et d'objets intelligents au sein d'un réseau dédié interconnecté. Ces objets, équipés de capteurs et d'actuateurs, deviennent des entités intelligentes capables de collecter, partager et traiter des données de manière autonome, ouvrant ainsi la voie à des applications innovantes dans divers domaines tels que la santé, l'industrie et la domotique. L'IoT crée un écosystème dynamique où les interactions entre les entités connectées vont au-delà du simple transfert de données, reflétant son impact profond sur la manière dont nous vivons, travaillons et interagissons avec notre environnement quotidien. Grâce à ses nombreux avantages, un réseau IoT est de plus en plus intégré dans des différents domaines d'application. Pour cela, dans le monde, le nombre d'appareils IoT connectés augmente d'une façon exponentielle visant à atteindre plus de 75 milliards d'appareils en 2025 [1] (Figure 2.1). Selon [5], environ 500 milliards d'appareils dotés de capteurs seront connectés à Internet d'ici 2030.
Avec l'augmentation du nombre d'objets IoT connectés, les réseaux IoT se complexifient davantage, nécessitant un déploiement et une organisation des composants selon une topologie spécifique. La section suivante énumère les topologies les plus courantes pour les réseaux IoT.

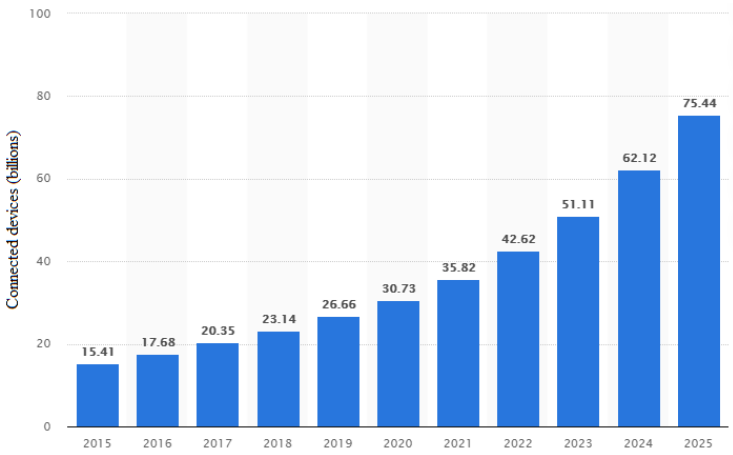


FIGURE 2.1 – Appareils IoT connectés de 2015 à 2025 [1]

2.2.1 Topologies des réseaux IoT

Les composants du réseau IoT sont structurés selon une topologie, dont le choix revêt une grande importance parmi les nombreux facteurs influant sur le bon fonctionnement et sur l'efficacité du réseau. Plusieurs topologies sont envisageables, parmi lesquelles figurent les plus communes :

— La topologie point à point (Figure 2.2(a)) impliquant des communications directes entre deux noeuds communicants ;

— la topologie en étoile (Figure 2.2(b)) permettant les communications seulement entre les noeuds terminaux et le point central ;

— la topologie maillée (Figure 2.2(d)) où les noeuds sont entièrement interconnectés, permettant la propagation des données à travers une architecture multi-sauts ;

— la topologie hybride (Figure 2.2(c)) consistant à connecter un ensemble de réseaux en étoile dans un réseau maillé ;

— la topologie en arbre (Figure 2.2(e)) organisant les composants de manière hiérarchique : les noeuds de chaque niveau sont connectés aux noeuds du niveau inférieur.

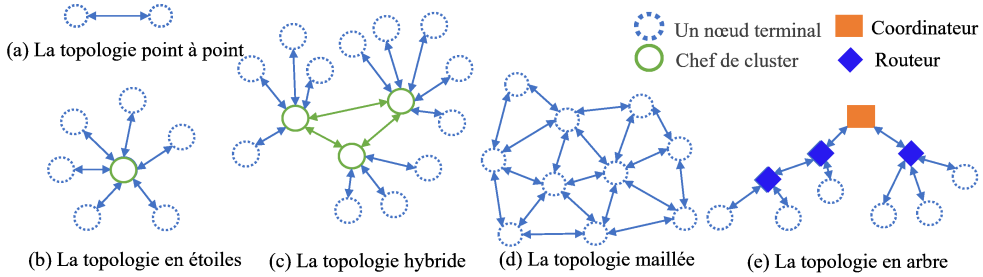


FIGURE 2.2 – Les topologies les plus courantes pour les réseaux IoT

Les données dans les environnements IoT doivent être gérées efficacement pour garantir le succès du système, en les traitant et les transmettant de manière fiable et rapide. Les technologies de communication jouent un rôle central dans cette transmission efficace, et elles évoluent continuellement pour répondre aux besoins changeants de l'IoT en tirant avantage des avancées technologiques dans les communications sans fil en particulier. La section suivante s'intéresse aux technologies de communication utilisées dans les réseaux IoT.

2.2.2 Technologies de communication pour les réseaux l'IoT

La connectivité joue un rôle crucial dans le succès des environnements IoT. Cependant, il n'y a pas de solution de communication universelle qui convienne à tous les cas d'utilisation et à toutes les applications IoT. Les réseaux IoT font appel à une variété de technologies de communication, incluant à la fois les connexions filaires comme l'Ethernet, HomeGrid/G.hn et Multimedia over Coax Alliance (MoCA) pour une stabilité et fiabilité accrues, et les technologies sans fil telles que le Wi-Fi, le Bluetooth et les réseaux cellulaires, pour offrir une connectivité flexible et mobile. Ces différentes technologies sont utilisées en fonction des besoins spécifiques des applications IoT permettant ainsi de répondre à une diversité d'exigences d'environnements. La Figure 2.3 présente une classification des technologies de communication employées dans les réseaux IoT, ainsi que des exemples illustratifs de ces technologies.

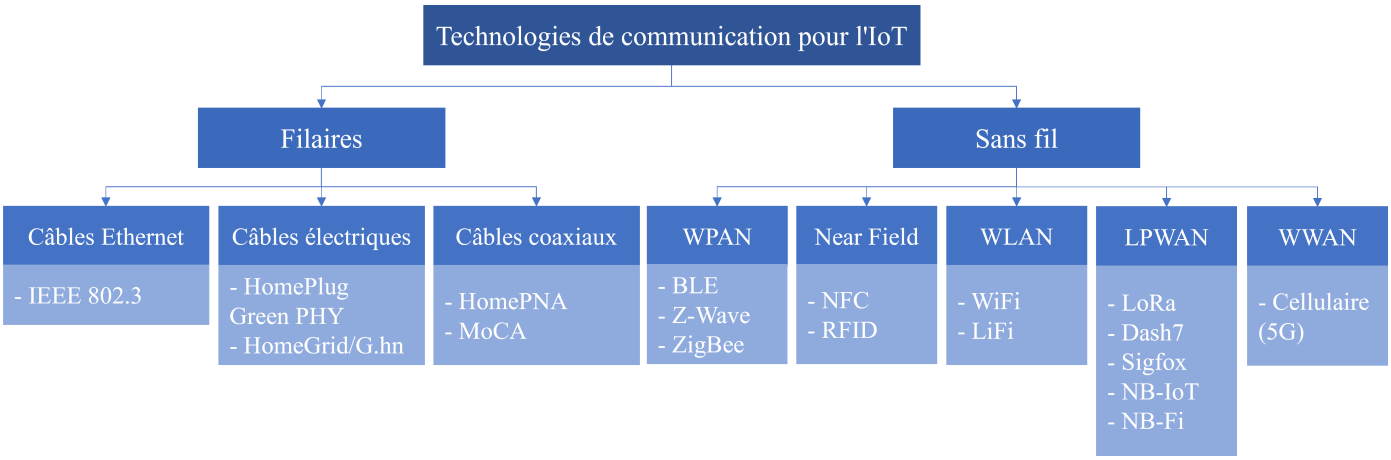


FIGURE 2.3 – Les technologies de communication utilisées dans les réseaux IoT

Nous constatons une grande diversité de technologies et de protocoles associés. Plusieurs tentatives d'uniformisation, comme MATTER [6], visent à les rendre interopérables. Les technologies de communication évoluent en permanence pour s'adapter aux exigences de l'IoT, favorisant une plus grande efficacité et succès des applications IoT. Grâce à cette réussite continue, une vaste gamme d'applications IoT a été conçue et mise en oeuvre, engendrant un impact majeur sur nos quoti-

diens personnels et professionnels. La section suivante présente une classification des domaines d'applications IoT et énumère quelques exemples.

2.2.3 Domaines d'applications de l'IoT

L'IoT offre une diversité de domaines d'application grâce à ses nombreux avantages, tels que la collecte de données en temps réel, la surveillance à distance et l'optimisation des processus. Cette technologie trouve des applications dans des secteurs variés où elle contribue à améliorer l'efficacité opérationnelle, à réduire les coûts et à répondre aux besoins évolutifs. La Figure 2.4 présente une classification des domaines d'application de l'IoT selon les principaux critères.

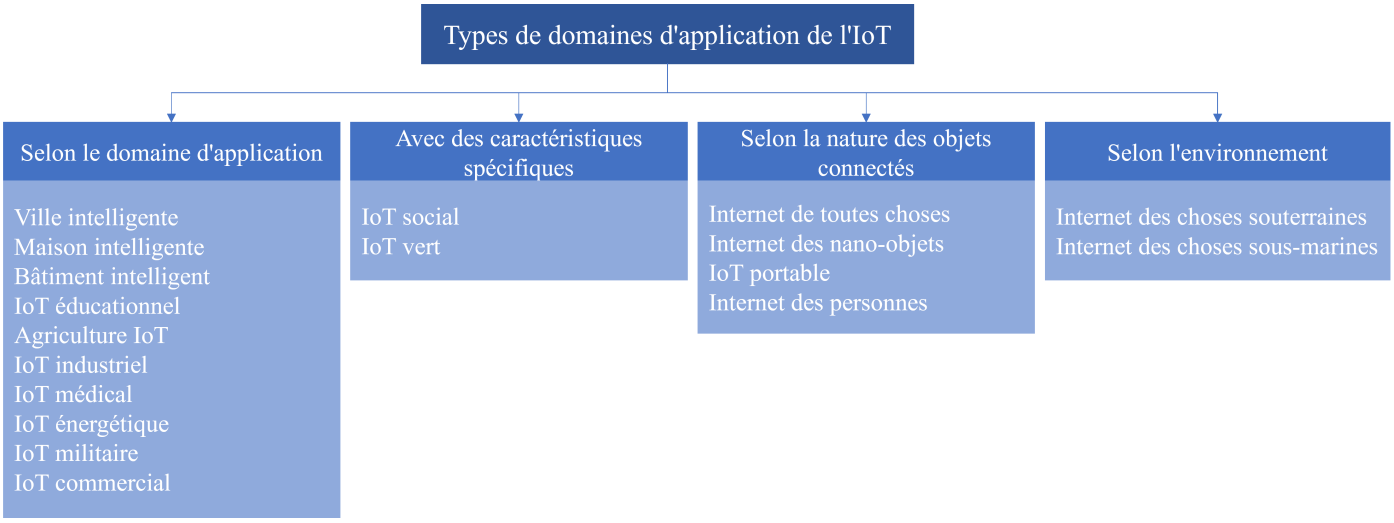


FIGURE 2.4 – Les principaux domaines d'applications de l'IoT

Bien que l'IoT rencontre des succès continus et s'étende à divers domaines d'application, il fait toujours face à de nombreux défis et problèmes. Les principaux défis de l'IoT sont présentés dans la section suivante.

2.2.4 Défis et enjeux de l'IoT

Les défis majeurs auxquels les applications IoT sont confrontées sont :

— La localisation en intérieur : déterminer la position des objets à l'intérieur des bâtiments ou des espaces fermés ;

— Le routage : sélectionner les chemins optimaux afin d'acheminer les données efficacement, en particulier entre les noeuds sans fil du réseau qui peuvent participer au routage de l'information sur la partie maillée de la topologie sans fil ;

— Une gestion efficace des données : une agrégation intelligente de données, une transmission rapide et fiable, un traitement rapide et la prise en considération de la variété des données gérées ;

— La sécurité et la confidentialité des données transmises et stockées ;

— La prise en considération de l'hétérogénéité des technologies utilisées ;

— La scalabilité des réseaux IoT et l'évolutivité des infrastructures ;

— L'optimisation de la consommation énergétique ;

— La sûreté de fonctionnement et une connectivité stable et résiliente ;

— Une réponse accrue aux attentes croissantes des consommateurs.

Cette section présente une vue d'ensemble des technologies de communication, des domaines d'application et des principaux défis de l'IoT. Notre publication [7] présente une revue décrivant ces sujets en détail. La section suivante se concentrera sur le routage dans un réseau IoT, soulignant ainsi son importance critique pour assurer une communication fluide.

2.3 Le routage dans les réseaux IoT

Dans le contexte des réseaux IoT, le routage fait référence au processus du choix des routes parcourues pour transférer les données entre les dispositifs connectés, tels que les capteurs, les actionneurs et les passerelles, à travers le réseau. Contrairement aux réseaux traditionnels, où les chemins de transmission sont souvent statiques et préétablis, le routage dans les réseaux IoT est dynamique et adaptable en raison de la nature changeante des dispositifs et de leur environnement. Cette dynamique exige des mécanismes de routage flexibles capables de s'ajuster aux variations des conditions du réseau, telles que la disponibilité des noeuds, les variations de la topologie et les contraintes de ressources. Un routage efficace revêt une importance cruciale dans les réseaux de l'IoT en raison de son impact direct sur la performance et la fiabilité des applications IoT. À cette fin, divers types de routage sont disponibles pour répondre aux exigences des différents environnements IoT. La sous-section suivante vise à décrire les principes, les avantages et les inconvénients des principaux types du routage.

2.3.1 Les types du routage

Les protocoles de routage pour les réseaux IoT peuvent être catégorisés selon plusieurs critères :

2.3.1.1 Selon la topologie du réseau

a **Le routage plat** : il se caractérise par une structure de réseau où tous les objets sont considérés comme équivalents. Contrairement à d'autres schémas de routage plus hiérarchiques, chaque objet IoT a les mêmes rôles et fonctionnalités. Ils collaborent pour accomplir la tâche globale du réseau en envoyant les données directement au noeud puits via un mode de communication multi-sauts (Figure 2.5).

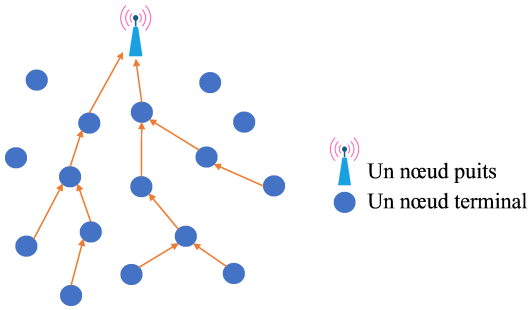


FIGURE 2.5 – Le routage plat

b **Le routage hiérarchique** : il repose sur la hiérarchie où les transmissions se font sur au moins deux niveaux. Les noeuds sont divisés en multiples sous-groupes, facilitant ainsi leur organisation et la gestion du réseau. La topologie la plus utilisée avec ce type de routage est la topologie en étoiles (Figure 2.6). Dans ce schéma, les objets IoT sont regroupés en clusters, et chaque cluster est supervisé par un noeud leader ou "chef de cluster". Ce dernier est responsable de l'agrégation des données provenant des objets membres du cluster avant leur envoi vers la destination finale. Le routage hiérarchique peut également être réalisé avec une topologie en arbre où les routeurs sont organisés en différents niveaux. Les routeurs de niveau inférieur transmettent généralement le trafic vers les routeurs de niveau supérieur pour un acheminement vers d'autres parties du réseau.

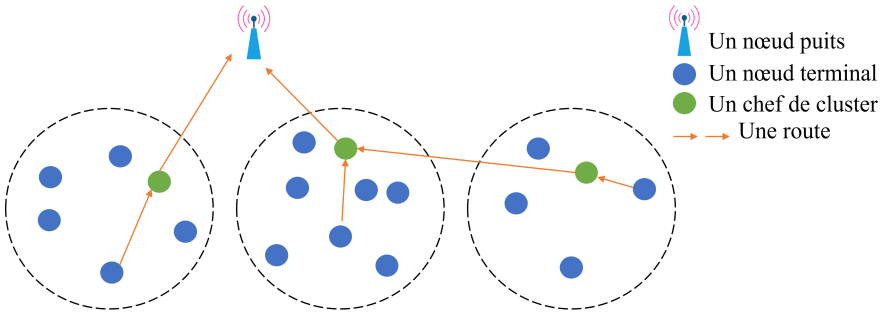


FIGURE 2.6 – Le routage hiérarchique dans une topologie en étoiles

c **Le routage géographique** : il repose sur l'utilisation des informations de localisation géographique des dispositifs pour optimiser les transmissions de données. Ce type de routage est basé sur l'estimation ou le calcul de la distance entre deux objets en fonction de leurs coordonnées spatiales, ce qui permet de déterminer le chemin le plus court ou plus généralement le plus efficace pour acheminer les données.

d **Le routage direct** : c'est un schéma de communication où les données sont transférées directement d'un noeud à un autre sans nécessiter de relais intermédiaire, selon une topologie totalement maillée (Figure 2.7(a)). Dans d'autres scénarios, il est possible que les objets soient situés à une distance de saut unique de la station de base ou du noeud central dans un routage direct (Figure 2.7(b)). Contrairement aux schémas de routage plus complexes qui impliquent des relais ou des noeuds intermédiaires, le routage direct simplifie la structure du réseau en établissant des communications directes entre les objets et avec la station de base.

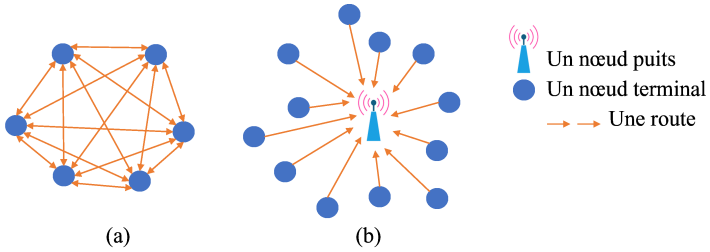


FIGURE 2.7 – Le routage direct

2.3.1.2 Selon le moment de détermination du chemin

a **Le routage proactif** : c'est une stratégie de gestion des communications où les chemins de transmission sont établis à l'avance et sont prêts à être utilisés dès que nécessaire. Le routage proactif anticipe les chemins optimaux entre les noeuds du réseau et les configure préalablement dans les tables de routage des dispositifs IoT. Face aux variations de topologie entre les objets connectés, dues au déplacement des noeuds et aux variations des conditions de propagation radio entre eux, ce type de routage n'est généralement pas bien adapté.

b **Le routage réactif** : c'est une méthode de gestion des communications où les routes ne sont établies que lorsqu'elles sont nécessaires et sont supprimées une fois l'acheminement des données à destination terminé. Contrairement au routage proactif qui préconfigure les routes à l'avance, le routage réactif répond de manière dynamique aux demandes d'acheminement de données en recherchant et en établissant des routes au moment de l'envoi des données.

c **Le routage hybride** : c'est une approche qui combine à la fois des éléments du routage réactif et du routage proactif pour optimiser les communications. Cette méthode tire parti des

avantages des deux paradigmes de routage en fonction des besoins spécifiques du réseau et des conditions changeantes de l'environnement IoT.

2.3.1.3 Selon le fonctionnement du protocole

a **Le routage basé sur la négociation** : c'est une méthode de gestion des communications qui repose sur l'échange de descripteurs de données entre les noeuds du réseau avant la transmission effective des données. Cette approche vise à réduire la redondance de l'information en permettant aux noeuds de négocier les détails de la transmission avant d'envoyer les données. Les descripteurs de données contiennent des informations sur les exigences de la transmission, telles que la qualité de service (Quality of Service (QoS)), les contraintes de bande passante et les préférences de routage.

b **Le routage multi-chemins** : c'est une stratégie de gestion des communications qui consiste à établir plusieurs chemins vers les destinations désirées. Cette approche vise à améliorer la fiabilité et la résilience du réseau en fournissant des routes alternatives en cas de défaillance ou de congestion sur le chemin principal. L'exemple extrême est l'inondation de tout le réseau, qui est malheureusement très couteux en énergie, ressources consommées, bande passante utilisée. Avoir plusieurs chemins possibles, dont au moins un chemin de secours, est une piste intéressante.

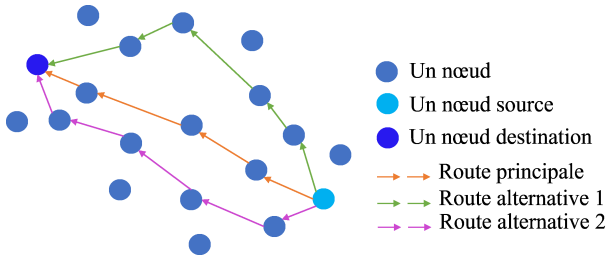


FIGURE 2.8 – Le routage multi chemins

c **Le routage basé sur la qualité des services (QoS)** : c'est une approche de gestion des communications qui vise à répondre à diverses exigences en matière de qualité de service telles que les délais de transmission, le niveau de fiabilité et la bande passante. Cette méthode de routage permet de garantir que les données sont acheminées en respectant des critères de performance prédéfinis pour répondre aux besoins spécifiques des applications IoT.

d **Le routage non cohérent** : c'est une approche de gestion des communications où les noeuds du réseau traitent localement les données brutes avant de les transmettre à des noeuds spécialisés pour un traitement ultérieur. Le routage non cohérent permet une distribution efficace du traitement des données en fonction de la spécialisation des noeuds du réseau.

e **Le routage cohérent** : c'est une stratégie de gestion des communications où les objets capteurs effectuent un minimum de traitement local sur les données brutes, tel que l'ajout de la date, le pointage de temps, ou la suppression des doublons, avant de transmettre ces données traitées à des objets spécialisés, souvent appelés agrégateurs, pour un traitement plus approfondi.

f **Le routage orienté-données** : il implique un processus en deux étapes (Figure 2.9) : tout d'abord, l'objet diffuse ses intérêts dans le réseau, spécifiant les données qu'il souhaite récupérer. Ensuite, les objets capteurs répondent aux requêtes en envoyant les données demandées à travers le chemin inverse de la requête jusqu'à l'objet demandeur.

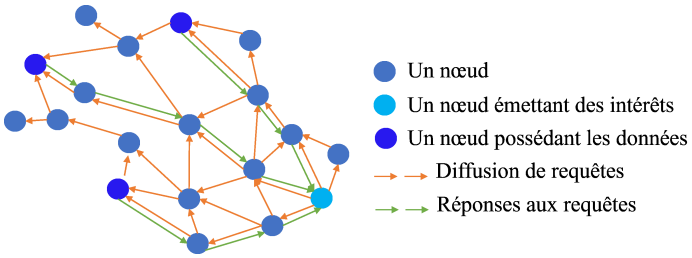


FIGURE 2.9 – Le routage orientée données

2.3.2 Synthèse

Le tableau 2.1 résume les principes, les avantages et les inconvénients de chaque type du routage.

TABLEAU 2.1 – La classification des protocoles de routage

Critère	Type	Description	Avantages	Inconvénients
La topologie du réseau	plat	Considérer tous les noeuds du réseau comme équivalents, permettant des communications directes sans hiérarchie de routage.	Simplicité, Faible latence, Évolutivité, Flexibilité	Scalabilité limitée, Manque de redondance, Vulnérabilité aux attaques, Consommation d'énergie plus élevée
	hiérarchique	Impliquer la division du réseau en niveaux de clusters où chaque cluster est géré par un noeud central appelé chef de cluster	Scalabilité, Redondance et une meilleure tolérance aux pannes, Gestion simplifiée	Complexité, Latence accrue, Surcharge du chef de cluster
	géographique	Utiliser les informations de localisation pour déterminer les chemins entre les noeuds communicants	Réduction de la latence, adaptabilité aux changements topologiques et aux mouvements des noeuds, aucun échange d'informations à l'exception de celles concernant les emplacements des voisins à un saut, un coût mémoire relativement faible, le passage à l'échelle est idéal, adressage simple, jointure simple du réseau, la localité	Besoin de localisation précise, Coût de localisation, Vulnérabilité aux attaques
	direct	Acheminer les données des objets capteurs vers une station de base en un seul saut	Faible latence, Simplicité, Efficacité énergétique	Portée limitée, Vulnérabilité aux pannes, Contraintes de densité
Le moment du détermination du chemin	proactif	Maintenir des chemins de communication préalablement définis entre les noeuds du réseau IoT, avant que des demandes de communication ne surviennent.	Faible latence, Gestion proactive, Stabilité	Surcharge de contrôle, Ressources consommées, Manque de flexibilité
	réactif	Établir des routes de communication uniquement en réponse à des demandes de communication	Utilisation efficace des ressources, Adaptabilité, Évolutivité	Latence accrue, Instabilité dans les réseaux dynamiques, Surcharge de contrôle et une utilisation excessive des ressources
	hybride	Combiner les approches proactives et réactives pour établir des routes de communication	Flexibilité, Équilibre entre performances et utilisation des ressources, Stabilité	Complexité de la coordination des deux méthodes de routage, Consommation de ressources
Le fonctionnement du protocole	basé sur la négociation	Utiliser des descripteurs de données pour échanger une série de messages de négociation avant la transmission effective des données	Élimination des informations redondantes, Adaptabilité aux besoins spécifiques des applications, Réduction de la charge de réseau	Complexité, Latence accrue, Besoin de synchronisation entre les noeuds du réseau
	multi-chemins	Établir plusieurs chemins de communication vers une destination	Tolérance aux pannes accrue, Équilibrage de charge, Amélioration des performances	Complexité (une coordination et une maintenance supplémentaires), Surcharge de contrôle, Coût de mise en oeuvre
	basé sur la qualité de service	Ajuster les chemins de communication en fonction des critères de qualité de service	Satisfaction des exigences de performance spécifiques, Utilisation efficace des ressources, Amélioration de la fiabilité	Complexité (surveillance des paramètres de QoS), Surcharge de contrôle, Besoin de QoS-awareness dans les noeuds
	non cohérent	Impliquer le traitement local des données brutes par les objets avant leur transmission à des noeuds spécialisés pour un traitement ultérieur.	Réduction de la charge du réseau, Autonomie des noeuds, Réduction de la latence	Perte de contexte ou de détails dans les informations transmises, Risque de redondance (pas de coordination entre les noeuds), Limitations de traitement local par les capacités des noeuds,
	cohérent	Impliquer un traitement initial minimal des données par les objets	Préservation du contexte, Utilisation efficace des ressources, Flexibilité de l'analyse	Augmentation de la latence, Besoin de coordination, Risque de surcharge
	orienté-données	Répondre aux demandes spécifiques de données en diffusant des intérêts dans le réseau	Utilisation ciblée des ressources, Adaptabilité aux besoins changeants du réseau, Réduction de la surcharge	Latence potentielle, Besoin de coordination entre les noeuds émetteurs d'intérêts et les noeuds détenteurs de données, Risque de perte de données

2.3.3 Comparaison et revue des études récentes sur le routage dans les réseaux IoT

Dans la littérature, diverses études se concentrent sur la résolution du problème du routage dans les réseaux IoT. Le tableau 2.2 présente une comparaison entre les approches récentes proposées pour aborder ce défi spécifique.

2.3.4 Synthèse

Dans cette section, nous avons introduit la notion du routage dans un réseau IoT, ses divers types et critères de performance, tout en décrivant les principales solutions de routage récentes. Dans notre publication [8], nous présentons une étude sur le routage dans les environnements IoT, fournissant ainsi plus de détails sur ce sujet. À partir de l'évaluation des travaux synthétisés, 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.
Pour mettre en oeuvre un algorithme de routage géographique dans un réseau IoT, il est essentiel d'adopter une solution de localisation, car le routage géographique repose sur des informations de localisation pour établir les chemins entre les noeuds communicants. Pour cela, la section suivante est consacrée à la localisation en intérieur dans les environnements IoT.

TABLEAU 2.2: Comparaisons entre des travaux récents résolvant le problème de routage dans les réseaux IoT

Papier	Année	Type	Approche/Objectifs	Techniques	Résultats	Inconvénients
[11]	2023	hiérarchique	une efficacité énergétique et une sécurité dans une topologie dynamique	sélection des chefs de clusters basée sur l'annonce des noeuds puits aux noeuds voisins, compression avec codage Huffman et chiffrement avec l'algorithme asymétrique RSA	consommation d'énergie : 1% de l'énergie initiale pourcentage de sécurité : 93%	Pas d'expérimentations réelles ; Les simulations sont réalisées en utilisant Matlab, plutôt qu'un simulateur spécifique aux réseaux IoT ; Le réseau simulé est limité à 5 noeuds.
[12]	2023	proactif, basé sur la QoS	Prendre en compte l'état actuel et futur du réseau et garantir un support de qualité de service différenciée.	"Graph Neural Network" et "Reinforcement Learning"	Packet Delivery Ratio (PDR) : 99% Débit : 60 paquets/sec Latence moyenne : 60ms	La stratégie de routage n'est pas capable de s'adapter efficacement en cas de changements topologiques, car elle nécessite un re-entraînement du modèle utilisé dans de telles situations ; Pas d'expérimentations réelles.
[13]	2023	basé sur la négociation	modéliser le problème du routage des véhicules comme un "Coalitional bargaining game" [14]	apprentissage par renforcement multi-agent	Réduction du temps d'exécution de 88% précision moyenne : 77% écart d'optimalité moyen : 3.9%	Le nombre d'agents est limité à 3 ; 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 ; Pas d'expérimentations réelles
[15]	2023	basé sur la QoS	minimiser la consommation d'énergie et améliorer la sécurité du routage	Ant Colony Optimization (ACO)	latence $\leq$ 14.5 ms amélioration de la consommation d'énergie de plus de 9.31% par rapport aux autres algorithmes comparés	Pas de noeuds mobiles ; Pas de noeuds hétérogènes ; Pas d'expérimentations réelles
[16]	2023	cohérent, basé sur la QoS	développer un mécanisme de routage pour les réseaux LoRa multi-sauts visant à acheminer les messages en tenant compte des exigences de QoS	le noeud LoRaute polyvalent agissant comme point d'accès et station de base	PDR : 96.81%	La performance de la stratégie proposée n'a pas été évaluée en utilisant les métriques standard de routage, et elle n'a pas été comparée à d'autres solutions de routage sur la base de ces critères.
[17]	2023	orienté-données	Diviser les données selon leur contenu et ajuster la topologie de routage en fonction de ce contenu	algorithme "Binary Gray Wolf Optimization" et logique floue	PDR : 98.35% débit 5700kbps	Pas de traitement de redondance de données ; Pas d'expérimentations réelles
[18]	2022	hiérarchique	une version de "Hierarchical Smart Routing Protocol" (HSRP) [19] en intégrant une nouvelle fonction de gestion de l'énergie	Smart Slumber : mettre en veille les chefs de clusters et les noeuds membres après chaque cycle pour une durée spécifiée	PDR : > 92% durée de vie : 7000 cycles	Pas d'expérimentations réelles

Suite à la page suivante

Tableau 2.2 – suite de la page précédente

Papier	Année	Type	Approche/Objectifs	Techniques	Résultats	Inconvénients
[20]	2022	plat, multi-chemins	une version de AODV [21] sécurisée contre les attaques "Blackhole"	établir plusieurs routes de la source à la destination pour contourner les chemins d'attaque	PDR : 85% Latence moyenne : 220ms Overhead : 60% Débit : 2.2 paquets/sec	La performance de la proposition est seulement comparée à celle de l'AODV, ce qui limite l'évaluation complète de son efficacité ; De plus, aucune expérimentation réelle n'a été réalisée ; Aucune modélisation mathématique n'est présentée.
[22]	2022	proactif	développer un protocole de routage adapté à la surveillance des Large Region of Interest (ROI) et visant à assurer une distribution équilibrée de la charge en matière de puissance de transmission.	un nouveau algorithme "Joint Decentralized Antenna" (JDA)	amélioration de 25 % de la durée de vie du réseau	Les résultats qui prouvent l'efficacité de la proposition et la comparaison avec d'autres solutions ne sont pas totalement présentés.
[23]	2022	géographique	optimiser le routage dans les Wireless Sensor Network (WSN) pour les applications multimédias en priorisant les chemins à haut débit plutôt que les chemins plus courts vers les stations de base.	éviter le routage persistant à travers le même chemin et mettre en oeuvre un contournement de zone vide du réseau	Latence moyenne : 5ms PDR : 97.25% énergie résiduelle : 99.7%	La proposition n'est pas évaluée en termes de débit qui est un critère clé pour les réseaux Multimédia ; Pas d'expérimentations réelles
[24]	2021	géographique	améliorer l'efficacité du routage géographique dans les réseaux de véhicules intelligents urbains (VANET)	utiliser un modèle de comportement dynamique des liaisons (Link Dynamic Behavior - LDB) pour évaluer la qualité des liaisons et sélectionner les chemins les plus fiables	PDR : 80%-98% Latence : 0.4s-0.7s Débit : 0.08Mbps-0.097Mbps	Pas d'expérimentations réelles
[25]	2021	basé sur la QoS	réaliser un routage efficace garantissant la QoS dans les "Software-defined Networks" (SDN)	divers algorithmes Machine Learning (ML) et une nouvelle méthode de classification appelée MACCA2-RF&RF	débit moyen : 208.07kbps latence moyenne : 106.52ms PDR : 99.48%	Pas de noeuds mobiles ; Pas d'expérimentations réelles
[26]	2020	géographique	limiter la surcharge de routage en maintenant des informations sur un seul voisin pour chaque noeud et équilibrer efficacement la consommation d'énergie des noeuds	appliquer un algorithme "mean square error" pour résoudre le problème de localisation des objets	PDR : 50% Délai : 3.6msec-5.5msec nombre de sauts moyen : 3.25 sauts	Problème de temps de calcul cumulatif pour acheminer les données vers la destination ; Comparaison avec un seul autre algorithme de routage ; Pas d'expérimentations réelles
[27]	2020	plat	une version améliorée de "Ad hoc On-Demand Distance Vector" (AODV)	exploiter la relation entre la portée de transmission et la densité pour ajuster dynamiquement l'utilisation de la puissance de transmission	PDR : 62% Latence moyenne : 730ms Jitter (J) : 0.063sec Débit : 150000 kbits/sec	La comparaison de la performance de la proposition avec uniquement AODV, ce qui ne permet pas d'évaluer pleinement l'efficacité de l'approche proposée ; Pas d'expérimentations réelles.

2.4 La localisation en intérieur dans les réseaux IoT

Dans le contexte croissant des applications IoT nécessitant une localisation précise, pour les environnements en extérieur, les systèmes GNSS sont couramment utilisés pour fournir une localisation en temps réel en extérieur. Cependant, en raison de leur faible capacité à pénétrer les environnements intérieurs, il est impératif de développer des solutions alternatives pour répondre aux besoins de localisation précise dans ces environnements spécifiques au sein des réseaux IoT. La localisation en intérieur se réfère à la capacité de déterminer avec précision la position des objets et des dispositifs à l'intérieur des bâtiments ou dans des espaces confinés. Ce défi spécifique est l'un des défis majeurs auxquels sont confrontés les déploiements d'IoT en raison de la complexité des environnements intérieurs. Ces environnements présentent des obstacles physiques tels que les murs, les plafonds et les planchers, qui entravent la transmission des signaux nécessaires à la localisation précise des appareils. Par conséquent, développer des solutions de localisation en intérieur efficaces et précises est essentiel pour garantir le bon fonctionnement des applications IoT dans ces environnements.

2.4.1 Les techniques de localisation en intérieur

Les solutions de localisation en intérieur sont souvent classées en deux catégories : range-based et range-free. La suite de cette sous-section définit et énumère les techniques de localisation en intérieur les plus courantes en se basant sur cette classification.

2.4.1.1 Les techniques de localisation range-free

Les techniques de localisation range-free se distinguent par leur capacité à fournir une estimation de position sans recourir à l'utilisation d'équipements de mesures puisqu'elles ne nécessitent pas de mesures de distance ou d'angle entre les noeuds. Elles reposent généralement sur des approches collaboratives, où les positions estimées sont dérivées des informations de voisinage et des méthodes géométriques. L'avantage majeur de ces techniques réside dans leur capacité à se passer d'infrastructures matérielles spécifiques, ce qui les rend économiquement plus avantageuses. Nous présentons, à la suite, des exemples des techniques range-free les plus répandues.

(a) **Centroid** : L'algorithme Centroid, proposé par [28] estime la position d'un noeud en se basant sur les informations de voisinage. Un exemple de l'application de Centroid est présenté dans la figure 2.10. Son principe consiste à suivre les étapes suivantes :

(i) Collecte des informations de voisinage : Soit N le nombre des voisins du noeud à localiser. Chaque voisin i transmet sa position estimée (x_i, y_i) au noeud à localiser. Les

noeuds voisins avec des positions connues servent d'ancres, fournissant des points de référence essentiels pour localiser le noeud cible.

(ii) Calcul de la position estimée $((\hat{x}, \hat{y}))$ selon l'équation 2.1 :

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

L'algorithme Centroid est relativement simple et ne nécessite pas de mesures de distance entre les noeuds. Cependant, sa précision dépend de la densité des ancres dans le voisinage et de la distribution spatiale des noeuds dans le réseau.

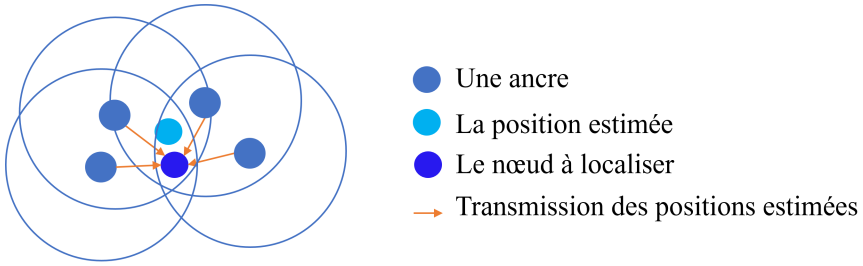


FIGURE 2.10 – Estimation de la localisation obtenue par l'algorithme Centroid

(b) **Distance Vector-Hop (DV-HoP)** : L'algorithme DV-HoP, proposé par [29] estime la position d'un noeud en se basant sur la distance à des ancres dont les positions sont connues. En effet, la distance entre un noeud et une ancre peut être estimée en fonction du nombre de sauts nécessaires pour atteindre l'ancre. Son principe de fonctionnement consiste à suivre les étapes suivantes :

(i) Les ancres diffusent leurs localisations et le nombre de sauts (incrémenté à chaque saut intermédiaire) dans tout le réseau ;

(ii) Chaque noeud récepteur i conserve H_{ij}, la valeur minimale reçue et l'utilise comme la distance la plus courte vers l'ancre j, en nombre de sauts ;

(iii) La distance moyenne par saut, d_{moy}, est calculée par les ancres et diffusée aux noeuds normaux voisins. Elle peut être calculée comme suit : distance entre deux ancres divisée par le nombre de sauts.

(iv) À partir des positions des ancres, les valeurs H_{ij} et d_{moy}, un noeud normal peut calculer sa position. La méthode de trilatération peut être appliquée, par exemple, pour estimer la position du noeud i. Cette méthode est expliquée dans (a).

L'algorithme DV-HoP offre une méthode de localisation économique et adaptable pour les réseaux sans fil, ne nécessitant pas de matériel spécialisé. Bien qu'il soit efficace

dans les réseaux denses, sa précision peut être compromise par l'effet de bruit et les erreurs de mesure, en particulier dans des environnements moins denses. Sa performance dépend également de la densité et de la disposition spatiale des ancres, ainsi que de la stabilité topologique du réseau.

De nombreuses variantes de DV-HoP ont été au cours des années, proposées par la communauté scientifique. CMWN-DVHop [30], WRCDV-Hop [31] et HW-DV-HopCSO [32] sont parmi les plus récentes.

(c) **Adaptive Point-In-Triangulation (APIT)** : L'algorithme APIT, proposé par [33], combine une approche adaptative avec le test Point-In-Triangulation (PIT) pour réduire la région estimative des noeuds à localiser, estimant ainsi la position sans indications de distances. Le principe de APIT consiste à :

(i) Soit A l'ensemble des ancres dans le réseau sans fil, dont les positions sont connues. Chaque noeud normal i collecte des messages balises provenant des ancres voisins ;

(ii) L'algorithme APIT réduit la région estimative R_i du noeud normal i en utilisant le test PIT avec différentes combinaisons de points d'ancrage voisins. Ce test est représenté par la figure 2.11 et l'équation 2.2 :

$$PIT_{ij} = \begin{cases} 1, & \text{si le noeud } i \text{ est à l'intérieur du triangle formé par les ancres } j \in A \\ 0, & \text{sinon} \end{cases} \tag{2.2}$$

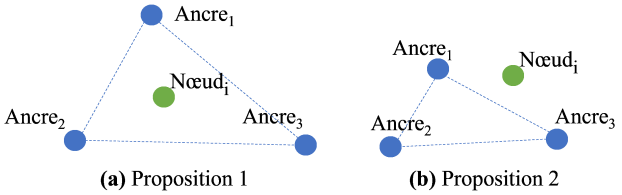


FIGURE 2.11 – Le test Point-In-Triangulation

(iii) Algorithme de balayage de grille : APIT utilise un algorithme de balayage de grille pour dériver la région d'intersection de tous les triangles formés par les points d'ancrage (figure 2.12). La région d'intersection est représentée par l'équation 2.3 :

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

(iv) Le centre de la région d'intersection Region_i est considéré comme l'emplacement estimé $(\hat{x}_i, \hat{y}_i)$ (figure 2.12).

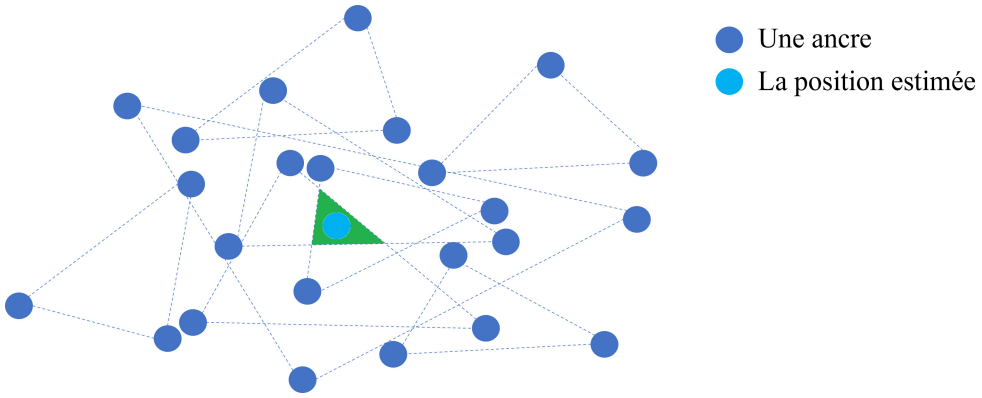


FIGURE 2.12 – L'algorithme APIT

Malgré sa précision accrue dans les environnements sans indications de distances, APIT peut souffrir de contraintes de mémoire et de calcul dues à la nécessité de diviser la zone du réseau en grilles, limitant ainsi son adaptation aux réseaux IoT.

(d) **Convex Position Estimation (CPE)** : L'algorithme CPE, proposé par [34], estime la position d'un noeud normal en utilisant les contraintes de connectivité avec ses ancres voisins pour définir une région de chevauchement, puis en calculant le centre du rectangle estimatif de cette région pour obtenir la position estimée du noeud normal. Le principe de CPE consiste à :

(i) Initialisation : Soit P l'ensemble des ancres dont les positions sont connues, où chaque ancre p_i est caractérisé par ses coordonnées (x_i, y_i) ;

(ii) Une matrice de connectivité C est construite, où $C_{ij} = 1$ si le noeud normal peut communiquer avec l'ancre j et $C_{ij} = 0$ sinon ;

(iii) L'algorithme CPE utilise les contraintes de connectivité pour estimer la région de chevauchement entre les régions de communication des ancres voisins ;

(iv) L'algorithme CPE définit le rectangle estimatif (Estimated Rectangle (ER)) qui borne la région de chevauchement (figure 2.13). Les quatre côtés du ER sont parallèles à l'axe des x et à l'axe des y ;

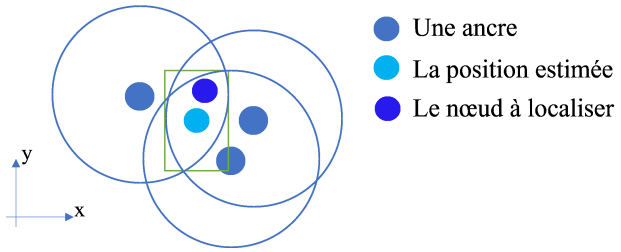


FIGURE 2.13 – L'algorithme CPE

(v) La position estimée du noeud normal est donnée par le centre du rectangle estimatif. Ce dernier peut être calculé en prenant la moyenne des coordonnées des ancres voisins impliqués dans la définition de la région de chevauchement.

Bien que l'algorithme CPE présente des avantages en termes de précision et de simplicité de mise en oeuvre, il est important de prendre en compte ses limitations liées à la dépendance à la connectivité, à la complexité de calcul et à la charge de communication lors de sa conception et de son déploiement dans un réseau sans fil.

2.4.1.2 Les techniques de localisation range-based

Les techniques de localisation range-based exploitent les mesures environnementales dans le contexte de l'IoT pour estimer la distance ou l'angle entre les appareils, offrant ainsi une estimation des positions. Leur mise en oeuvre nécessite généralement l'utilisation de matériel plus coûteux et peut impliquer un déploiement plus complexe en raison de la nécessité de calibrer et de maintenir les équipements de mesure. Cependant, ces techniques peuvent être particulièrement adaptées aux applications IoT exigeant une précision élevée, telles que la surveillance environnementale de précision ou la navigation à l'intérieur des bâtiments. Parmi les méthodes de mesure couramment citées dans la littérature figurent celles décrites dans [35]. Les techniques de mesures les plus utilisées sont :

(a) **Received Signal Strength Indicator (RSSI)** [36] : est une mesure de la force du signal reçu lors de transmissions sans fil, utilisée pour estimer la distance entre l'émetteur et le récepteur. Elle ne nécessite aucun équipement supplémentaire et est largement utilisée pour la localisation en intérieur. Cependant, sa précision peut être affectée par des obstacles et des antennes non omnidirectionnelles.

(b) **Channel State Information (CSI)** : combine des informations sur l'état du canal de communication décrivant la propagation du signal de l'émetteur au récepteur. Selon [37], le CSI offre une meilleure précision que le RSSI en matière de détection, bien qu'il soit moins performant que la technique de mesure du temps de vol (ToF) radio. Cette technique est bien adaptée à des liaisons Wi-Fi pour lesquelles les transceivers radio offrent généralement cette information de CSI.

(c) **Angle of arrival (AoA)** : mesure l'angle auquel le signal atteint le récepteur. C'est une technique assez récente qui est principalement proposée par des tranceivers radio multi-antennes UWB et plus récemment Bluetooth pour les réseaux IoT.

(d) **ToF** : mesure le temps nécessaire pour qu'un signal atteigne la destination. En utilisant la vitesse de propagation du signal radio, cette mesure permet d'estimer la distance entre deux objets. Les signaux UWB permettent des mesures ToF précises, utilisées dans des solutions

de localisation visant une grande précision. Face à la difficulté d'avoir une horloge commune entre émetteur et récepteur, on utilise souvent l'aller et retour radio (Two-Way Ranging (TWR)) [38] pour mesurer le temps de vol.

(e) **Time Difference of Arrival (TDoA)** : une variante du ToF qui mesure la différence de temps d'arrivée d'un signal provenant d'un émetteur vers au moins deux récepteurs distincts considérés comme des points de référence. En combinant cette mesure avec la vitesse de transmission du signal, la différence de distance entre l'émetteur et les deux récepteurs peut être calculée [39]. Avec 3 récepteurs ancres dont on connait la position, on peut évaluer la position de l'émetteur mobile.

(f) **Signal-to-Noise Ratio (SNR)** : mesure la qualité d'une transmission en comparant le niveau du signal au niveau du bruit. Elle est définie comme le rapport entre la puissance du signal de sortie souhaité et la puissance du bruit, souvent exprimée en décibels [40].

À partir de ces mesures, il faut ensuite mettre en place des algorithmes de localisation. Généralement, ils se basent sur les techniques de trilatération ou de triangulation qui sont des méthodes de détermination de la position d'un objet ou d'un dispositif en utilisant la mesure de distances ou d'angles à partir de plusieurs points de référence connus. Les méthodes les plus couramment utilisées sont :

(a) **La trilatération** : Cette méthode utilise les mesures de distance entre les noeuds pour déterminer la position en utilisant des algorithmes de trilatération, tels que la méthode des cercles ou la méthode des hyperboles. Par exemple en appliquant la méthode des cercles, l'une des mesures mentionnées précédemment est utilisée pour déterminer la position de l'objet par l'intersection de cercles (ou de sphères en 3D) centrés sur les ancres, où les rayons sont déterminés par les mesures de distances. Seulement trois mesures sont nécessaires et utilisées en 2D (et seulement quatre en 3D). Un exemple en 2D est illustré dans la figure 2.14. Si les coordonnées des 3 ancres sont respectivement, $(x1, y1)$, $(x2, y2)$ et $(x3, y3)$, et r_1, r_2 et r_3 sont les rayons des trois cercles, l'équation qui calcule la position estimée de l'objet à localiser est le système d'équations non linéaires résultant de ces trois équations de cercle (équation 2.4).

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

La méthode de localisation par trilatération offre une approche simple et pratique basée sur la mesure des distances entre des ancres et l'objet à localiser, ce qui la rend adaptée à divers environnements. Cependant, elle peut être sensible aux erreurs de mesure, aux obstacles et nécessite un nombre important d'ancres pour assurer une couverture adéquate, ce qui peut

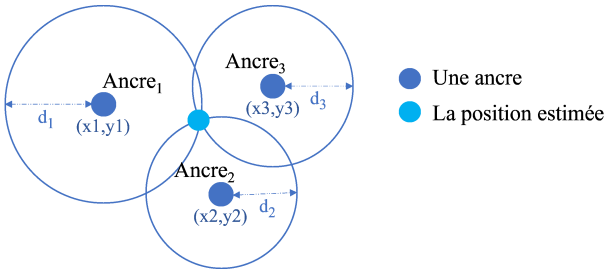


FIGURE 2.14 – Principe de la trilatération

augmenter les coûts de déploiement et limiter sa précision dans certains environnements. Le calcul théorique est souvent impossible à cause des petites erreurs de distances dans la réalité. Il existe alors de nombreux algorithmes (MinMax [41] étant le plus simple) pour contourner le problème.

(b) **La multilatération** : Similaire à la trilatération mais utilisant plus de trois mesures de distance pour améliorer la précision de la localisation. Considérons un exemple de multilatération avec les hyperboles en utilisant n ancres. Les coordonnées des ancres sont données par (x_i, y_i) pour $i = 1, 2, \ldots, n$. La distance mesurée entre l'objet à localiser et chaque ancre est notée d_i. La distance de référence ou le temps de vol de référence est d_{ref}. Les mesures de référence sont généralement déterminées lors de la phase d'installation ou de configuration du système de localisation. Ces valeurs servent de référence pour calibrer les mesures de distances ou les temps de vol obtenus à partir des signaux émis par les ancres vers l'objet. En utilisant les mesures de distances, les équations des hyperboles peuvent être formulées comme suit (équation 2.5) :

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

où c_i représente le biais de mesure associé à l'ancre i. Ce biais peut être dû à des erreurs de mesure ou à des facteurs environnementaux perturbateurs, et il est généralement pris en compte pour corriger les mesures de distance.

La localisation par multilatération offre une grande précision et une résistance accrue aux interférences. Cependant, elle peut être sensible aux retards de propagation et aux changements environnementaux, ce qui peut limiter sa précision dans des conditions complexes.

(c) **La localisation par prédiction par Machine Learning** : Cette technique utilise des algorithmes ML pour estimer la position géographique d'un objet à l'intérieur. Elle repose sur l'analyse des données collectées par des capteurs pour prédire la position en temps réel, offrant ainsi une méthode précise et flexible pour la localisation dans divers domaines d'application. Cette technique implique généralement les étapes suivantes :

(i) Construire un dataset complet pour entraîner le modèle en collectant des données d'apprentissage à partir d'objets, y compris des mesures telles que la puissance du signal, le ToF ou les caractéristiques environnementales, ainsi que leurs positions connues ;

(ii) Un modèle ML, tel qu'un modèle DL, un arbre de décision, ou une méthode de régression, est entraîné sur les données d'entraînement pour apprendre la relation entre les caractéristiques des données et les positions de localisation correspondantes. Le modèle est ajusté itérativement pour minimiser l'erreur entre les prédictions et les positions réelles ;

(iii) Validation du modèle : Le modèle entraîné est validé à l'aide de données de validation pour évaluer ses performances sur des données non vues ;

(iv) Une fois que le modèle est entraîné et validé, il peut être utilisé pour prédire la position de localisation d'un appareil ou d'un objet en utilisant de nouvelles données en temps réel. Ces données sont introduites dans le modèle et le modèle génère une estimation de la position en fonction des caractéristiques observées.

La localisation par prédiction par ML offre une grande précision et flexibilité, mais nécessite souvent beaucoup de données d'entraînement et peut être sujette à des erreurs si les modèles ne sont pas correctement entraînés ou si les conditions environnementales changent. La mise en oeuvre peut également nécessiter des ressources informatiques importantes pour entraîner le modèle.

(d) **Le fingerprinting** : Cette méthode utilise des empreintes de signal (fingerprint) pour associer des emplacements connus à des motifs de signal spécifiques, permettant ainsi de déterminer la position de l'objet en fonction des mesures de signal reçues. L'application de la méthode fingerprinting consiste aux étapes suivantes :

(i) La collecte des empreintes digitales impliquant la construction d'une base de données de signal radio à partir de différents points dans la zone à localiser ;

(ii) Un algorithme d'apprentissage automatique ou une méthode de traitement du signal est utilisé pour créer un modèle ou une carte de signal à partir des données stockées dans la base de données ;

(iii) Lorsqu'un appareil souhaite être localisé, il mesure les signaux radio disponibles à son emplacement actuel. Ces données de signal sont ensuite comparées au modèle ou à la

carte de signal créé lors de la phase d'apprentissage. En trouvant la correspondance la plus proche entre les données de signal mesurées et le modèle de signal dans la base de données, l'emplacement de l'appareil est estimé.

Les avantages de la méthode de localisation fingerprinting incluent une haute précision et une adaptabilité à différents environnements, mais elle nécessite une phase d'apprentissage intensive pour créer la base de données souvent volumineuse et peut être sensible aux variations de l'environnement radiofréquence.

(e) **La triangulation angulaire** : Cette méthode estime la position d'un objet en se basant sur les angles de réception de signaux provenant de plusieurs ancres connues réparties dans l'environnement. Supposons que l'environnement contient n ancres avec des coordonnées (x_i, y_i) pour $i = 1, 2, ..., n$. Les mesures des angles θ_i entre chaque ancre et l'objet à localiser doivent être prises (figure 2.15).

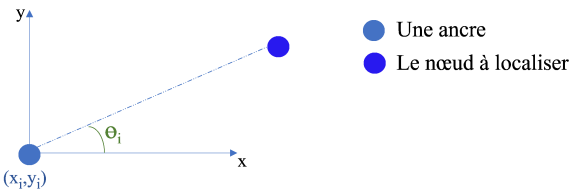


FIGURE 2.15 – Mesure de l'angle entre une ancre i et l'objet à localiser

Il est généralement nécessaire de disposer de deux antennes dans le même noeud récepteur pour qu'il puisse mesure l'angle d'arrivée du signal radio par évaluation de la différence d'instant d'arrivée ou de déphasage de la modulation radio reçue. Pour réaliser une localisation bidimensionnelle par triangulation angulaire, il est impératif d'avoir au moins deux ancres. Le fonctionnement de ce principe est démontré dans la figure 2.16. La position de l'objet à localiser peut être déterminée en utilisant les deux angles θ_1 et θ_2, en appliquant l'équation 2.6 :

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

Un autre scénario est possible : un noeud dont la position est connue, capable de mesurer une distance et un angle par rapport à un mobile, est donc en mesure de le localiser.
Bien que la localisation par triangulation angulaire est efficace dans de nombreuses applications où les bonnes mesures des angles peuvent être garanties en offrant une bonne précision, elle nécessite des dispositifs de mesure d'angle précis et peut être sensible aux erreurs d'orientation ou de cali-

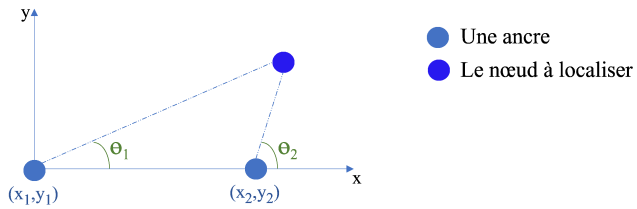


FIGURE 2.16 – La triangulation angulaire avec deux ancres

bration. Les objets connectés étant souvent très petits, la distance entre les 2 antennes d'un même récepteur est faible, ce qui limite la précision de la mesure d'angle.

2.4.2 Comparaison et revue des études récentes sur la localisation en intérieur

Dans la littérature, plusieurs études se focalisent sur la résolution du problème de localisation en intérieur dans les réseaux IoT. Le tableau 2.3 offre une comparaison entre des approches récentes proposées pour traiter ce défi particulier.

2.4.3 Synthèse

L'examen des travaux susmentionnés a montré 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, et 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. Quant aux autres travaux, la précision obtenue ne satisfait pas aux exigences de localisation intérieure pour les applications IoT. 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.

TABLEAU 2.3: Comparaisons entre des travaux récents résolvant le problème de localisation en intérieur dans les réseaux IoT

Papier	Année	Type	Approche/Objectifs	Techniques	Technologie	Mesures	Résultats		Inconvénients
							Mean Absolute Error (MAE)	Précision	
[43]	2023	range-based	prédire les positions à l'intérieur d'un bâtiment universitaire	Deep Neural Network (DNN)	LoRaWAN	RSSI, SNR	-	98.8%	La proposition n'a pas été comparée à d'autres solutions utilisant les mesures RSSI et SNR
[30]	2023	range-free	Développer CMWN-DVHop, une version améliorée de l'algorithme DV-HoP en introduisant des facteurs de segmentation et de pondération.	"Cyclotomic method" et l'algorithme "Weighted Recursive Least-Squares (WRLS)"	-	-	2.9m	-	Pas d'expérimentations réelles ; Les simulations sont effectuées à l'aide de Matlab, plutôt qu'avec un simulateur dédié aux réseaux IoT.
[31]	2023	range-free	Proposer WRCDV-Hop qui comporte quatre améliorations de l'algorithme DV-HoP	Une méthode pondérée, Whale Optimization Algorithm (WOA)	-	-	0.13m	-	Les résultats sont basés sur des simulations plutôt que sur des données réelles ; Les simulations sont effectuées en utilisant Matlab, plutôt qu'un simulateur spécifique aux réseaux IoT.
[32]	2023	range-free	Proposer un algorithme de localisation, HW-DV-HopCSO, une version améliorée de DV-HoP	Chicken Swarm Optimization (CSO)	-	-	-	80%	La précision obtenue ne répond pas aux exigences des applications IoT ; Les résultats sont obtenus à partir des simulations utilisant Matlab, plutôt qu'un simulateur spécifique aux réseaux IoT.
[44]	2023	range-free	Optimiser l'algorithme de localisation APIT en améliorant le prétraitement des données RSSI	Gaussian filtering, Artificial Neural Network (ANN)	-	RSSI	1.55m	-	L'erreur de distance moyenne obtenue ne satisfait pas aux exigences des applications IoT ; Les résultats sont issus des simulations effectuées à l'aide de Matlab au lieu d'un simulateur dédié aux réseaux IoT
[45]	2022	range-based	Développer deux solutions de localisation dans des espaces intérieurs	Support Vector Regression (SVR) et Kalman Filter (KF)	-	RSSI	0.85m	-	Pas d'expérimentations réelles
[46]	2022	range-based	Développer un framework, CSILoc, qui inclut une solution DL visant à obtenir une estimation précise de la localisation	Fingerprinting, Convolutional Neural Network (CNN)	Wi-Fi	CSI	1.75m	68.5%	Les résultats obtenus ne répondent pas aux exigences de localisation intérieure de l'IoT.

Suite à la page suivante

Tableau 2.3 – suite de la page précédente

Papier	Année	Type	Approche/Objectifs	Techniques	Technologie	Mesures	Résultats		Inconvénients
							MAE	Précision	
[47]	2022	range-free	Proposer une version améliorée de l'algorithme "Triangle Centroid Localization"	PIT	-	RSSI	1.42m	-	Les résultats reposent sur des simulations réalisées à l'aide de Matlab, plutôt que sur des données réelles, et ne sont pas issus d'un simulateur spécifique aux réseaux IoT ; L'erreur de distance moyenne obtenue ne satisfait pas aux critères de localisation intérieure requis pour l'IoT.
[48]	2021	range-based	Proposer un nouveau modèle d'ensemble comme solution de localisation intérieure	Fingerprinting, KNN, DNN et RF	Wi-Fi	RSSI	1.10m	-	La moyenne des erreurs de distance obtenue n'était pas adéquate pour relever le défi de la localisation intérieure dans les environnements IoT.
[49]	2021	range-based	Élaborer une image 2D pour servir de base à un modèle DL, puis appliquer une variante améliorée de l'algorithme Particle Swarm Optimization (PSO) afin d'optimiser son entraînement.	CNN et une version améliorée de PSO	BLE	RSSI	-	97.92%	La précision obtenue est fondée sur les résultats de la phase de validation. Toutefois, leur fiabilité doit être vérifiée à travers une étape de test distincte pour confirmer leur validité.
[50]	2021	range-based	Développer une solution de localisation intérieure (Poucet) adaptée aux besoins des pompiers et des soldats.	Trilatération	UWB, LoRa	ToF	0.36m	-	Pas d'expérimentations réelles
[51]	2021	range-based	Développer une solution de localisation de véhicules aériens sans pilote (Unmanned Aerial Vehicle - UAV) pour des applications dans des environnements extrêmement confinés.	Méthode Maximum Likelihood Estimation (MLE)	UWB	ToF	0.2m	-	Les performances de la solution proposée ne sont pas évaluées par rapport à d'autres solutions existantes pour démontrer son efficacité.
[52]	2021	range-based	Proposer un système de positionnement intérieur pouvant fonctionner sur n'importe quel smartphone grâce à une complexité temporelle réduite	Fingerprinting, un nouvel algorithme "Particle Filter" proposé	BLE	RSSI	1.4m	-	L'efficacité de la solution proposée n'est pas comparée à celle d'autres solutions existantes pour évaluer ses performances.
[53]	2020	range-based	Proposer une nouvelle solution de localisation et de suivi d'objets en intérieur	Fingerprinting, Multi-layer Perceptron (MLP)	Wi-Fi	RSSI	-	83%	Les résultats doivent être validés expérimentalement sur des données collectées dans un environnement réel.

Suite à la page suivante

Tableau 2.3 – suite de la page précédente

Papier	Année	Type	Approche/Objectifs	Techniques	Technologie	Mesures	Résultats		Inconvénients
							MAE	Précision	
[54]	2020	Range-free	Développer et présenter une solution de localisation distribuée et collaborative	Un nouvel algorithme Uncertainty Weighted Localization (UWL)	UWB	-	<1m	-	Il est essentiel de réaliser des expériences pratiques afin de confirmer la validité de ces résultats.
[55]	2020	range-based	Proposer une solution de localisation en intérieur appelée Deep Learning-Based Cooperative Architecture (DELTA)	Fingerprinting, DNN	5G	RSSI	1.6m	89%	La performance du modèle proposé doit être vérifiée dans des conditions réelles et au cours d'une phase de test supplémentaire, et non seulement par le biais d'une validation ; La précision obtenue ne répond pas aux exigences des réseaux IoT pour la localisation intérieure.

2.5 Conclusion

En conclusion, ce premier chapitre a exploré deux défis clés liés aux réseaux IoT : le routage et la localisation en intérieur. Nous avons examiné les différents aspects du routage, y compris les types, les indicateurs de performance et les études récentes, ainsi que les aspects clés de la localisation en intérieur, notamment les techniques, les critères d'évaluation et les recherches actuelles. Ce chapitre a permis de mettre en évidence l'importance du routage efficace et de la localisation précise pour un fonctionnement optimal dans les réseaux IoT. Dans le chapitre suivant, nous explorerons les techniques d'optimisation et de DL, en examinant comment ces approches peuvent être appliquées pour résoudre les défis spécifiques rencontrés dans les réseaux IoT, notamment en termes de routage et de localisation en intérieur.

Chapitre 3

Optimisation et Deep Learning

Sommaire

3.1 Introduction

Ce chapitre explore deux domaines offrant des outils puissants pour résoudre une multitude de problèmes complexes. Nous commençons par examiner les types de problèmes d'optimisation, mono-objectifs et multi-objectifs. Nous explorons également les notions essentielles qui sont fondamentales pour comprendre les méthodes de résolution des problèmes d'optimisation. Ensuite, nous examinons les méthodes de résolution des problèmes d'optimisation, en mettant particulièrement l'accent sur les métaheuristiques et les méthodes hybrides. Par la suite, nous explorons les critères utilisés pour évaluer ces méthodes. La première partie est terminée par une revue des méthodes récentes de résolution des problèmes d'optimisation. Dans la seconde partie de ce chapitre, nous nous intéressons au DL. Nous examinons les principaux domaines d'application du DL, avec un accent particulier sur son utilisation pour la régression. Nous explorons également les fondements théoriques du DL, y compris le fonctionnement des techniques d'entraînement et d'optimisation des paramètres. Ensuite, nous nous intéressons aux principales techniques de DL qui peuvent être appliquées pour les problèmes de régression. À la suite, les critères de performance utilisés pour évaluer ces techniques sont abordées. La dernière partie de ce chapitre sert à souligner les avancées récentes et les tendances de recherche dans ces domaines passionnants, ainsi que leur utilisation pour résoudre les problèmes de routage et de localisation dans les réseaux IoT.

3.2 Les problèmes d'optimisation

Les problèmes d'optimisation sont des situations où il est nécessaire de trouver la meilleure solution, souvent en maximisant ou minimisant une ou plusieurs fonctions objectifs, parmi un ensemble de solutions possibles, généralement définies par un espace de recherche et des contraintes. Les problèmes d'optimisation se retrouvent dans de nombreux domaines, tels que l'ingénierie, l'économie, la logistique, la finance, et bien d'autres, et leur résolution efficace est cruciale pour la prise de décision et l'amélioration des processus dans ces domaines. Ces problèmes peuvent être classés en plusieurs catégories en fonction de la nature de la fonction objectif et des contraintes. Nous nous intéressons, dans cette première section, à la classification selon le nombre d'objectifs.

3.2.1 Les problèmes d'optimisation mono-objectifs

Les problèmes d'optimisation mono-objectifs sont des problèmes dans lesquels une seule fonction objectif doit être maximisée ou minimisée, sous réserve de certaines contraintes. Mathématiquement, un problème d'optimisation mono-objectif peut être formulé comme suit (équation 3.1) :

$$\begin{aligned} \text{Minimiser/Maximiser} \quad & f(x) \\ \text{sous contraintes} \quad & g_i(x) \leq 0, \quad i = 1, 2, \ldots, m \\ & h_j(x) = 0, \quad j = 1, 2, \ldots, p \end{aligned} \tag{3.1}$$

où $f(x)$ est la fonction objectif à optimiser, x est le vecteur des variables de décision, $g_i(x)$ sont les contraintes d'inégalité, et $h_j(x)$ sont les contraintes d'égalité. L'objectif est de trouver la valeur optimale du vecteur x qui minimise (ou maximise) la fonction objectif $f(x)$ tout en satisfaisant les contraintes $g_i(x)$ et $h_j(x)$.

3.2.2 Les problèmes d'optimisation multi-objectifs

Les problèmes d'optimisation multi-objectifs (Multi-objective Optimization Problems (MOP)) sont des problèmes dans lesquels plusieurs fonctions objectifs doivent être optimisées simultanément, souvent en conflit les unes avec les autres. Mathématiquement, un problème d'optimisation multi-objectifs peut être formulé comme suit (équation 3.2) :

$$\begin{aligned} \text{Minimiser/Maximiser} \quad & f_i(x), \quad i = 1, 2, \ldots, m \\ \text{sous contraintes} \quad & g_j(x) \leq 0, \quad j = 1, 2, \ldots, n \\ & h_k(x) = 0, \quad k = 1, 2, \ldots, p \end{aligned} \tag{3.2}$$

où $f_i(x)$ sont les fonctions objectifs à optimiser, x est le vecteur des variables de décision, $g_j(x)$ sont les contraintes d'inégalité, et $h_k(x)$ sont les contraintes d'égalité. L'objectif est de trouver le vecteur de décision x qui minimise (ou maximise) simultanément toutes les fonctions objectifs $f_i(x)$ tout en satisfaisant les contraintes $g_j(x)$ et $h_k(x)$.
Afin de simplifier la rédaction dans la suite de cette section, nous supposons qu'un problème d'optimisation consiste à un problème de minimisation.

3.2.3 Les notions principales

Soit S l'ensemble des solutions réalisables et $F(X)$ l'ensemble des fonctions objectif à optimiser dans un problème d'optimisation multi-objectifs (avec m objectifs). Une solution réalisable est une solution qui se trouve dans la région réalisable du problème et satisfait toutes ses contraintes. Elle n'a pas nécessairement à être optimale, mais elle doit respecter les règles imposées par les contraintes.

3.2.3.1 La dominance

La notion de dominance est utilisée dans les problèmes d'optimisation multi-objectifs pour comparer les solutions entre elles. Une solution x_1 est dite dominante par rapport à une autre solution x_2 si elle offre des valeurs meilleures ou égales que x_2 pour tous les objectifs et strictement meilleure dans au moins un objectif. Formellement, soit $f(x_1) = (f_1(x_1), f_2(x_1), ..., f_m(x_1))$ et $f(x_2) = (f_1(x_2), f_2(x_2), ..., f_m(x_2))$ les vecteurs objectifs associés aux solutions x_1 et x_2 respectivement, x_1 domine x_2 si et seulement si :

$$\begin{aligned} & f_i(x_1) \leq f_i(x_2) \quad \text{pour tout} \quad i \in \{1, 2, ..., m\} \\ & \text{et} \\ & \exists\, i \text{ tel que} \quad f_i(x_1) < f_i(x_2) \end{aligned} \tag{3.3}$$

Cette dominance est exprimée par : $x_1 \succ x_2$. La figure 3.1 contient un exemple de notion de dominance où : l'étoile est dominée par les cercles ; l'étoile domine les carrés ; l'étoile est équivalente aux triangles au sens de la dominance.

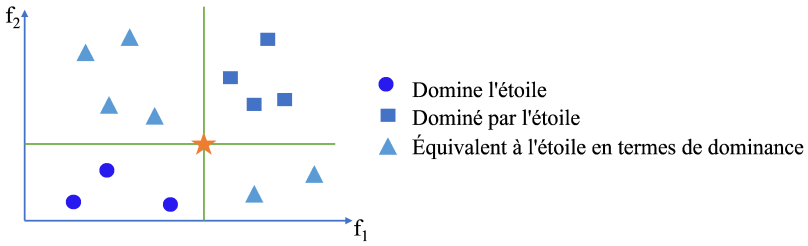


FIGURE 3.1 – Un exemple de notion de dominance

Lorsqu'un problème d'optimisation comporte des contraintes, la notion de dominance doit être adaptée pour les prendre en compte. Une solution x_1 est considérée comme dominante par rapport à une autre solution x_2 si l'une des trois situations suivantes est vraie :

— x_1 et x_2 satisfont toutes les contraintes et x_1 domine x_2. Formellement :

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

— x_1 satisfait toutes les contraintes tandis que x_2 n'en satisfait pas toutes. Formellement :

$$\begin{cases} g_j(x_1) \leq 0 \quad \text{pour tout} \quad j = 1, 2, \ldots, n \quad \text{et} \quad h_k(x_1) = 0 \quad \text{pour tout} \quad k = 1, 2, \ldots, p \\ \exists\, j \text{ tel que} \quad g_j(x_2) > 0 \quad \text{ou} \quad \exists\, k \text{ tel que} \quad h_k(x_2) \neq 0 \end{cases} \tag{3.5}$$

— x_1 et x_2 ne satisfont pas toutes les contraintes, mais x_1 présente moins de violations de contraintes que x_2. Formellement :

$$\begin{cases} \exists\, u_1 \text{ tel que} \quad u_1 \subset \{1, 2, \ldots, n\} \quad \text{et} \quad g_j(x_1) > 0 \quad \text{pour tout} \quad j \in u_1 \\ \text{et/ou} \quad \exists\, v_1 \text{ tel que} \quad v_1 \subset \{1, 2, \ldots, p\} \quad \text{et} \quad h_k(x_1) \neq 0 \quad \text{pour tout} \quad k \in v_1 \\ \text{et} \\ \exists\, u_2 \text{ tel que} \quad u_2 \subset \{1, 2, \ldots, n\} \quad \text{et} \quad g_j(x_2) > 0 \quad \text{pour tout} \quad j \in u_2 \\ \text{et/ou} \quad \exists\, v_2 \text{ tel que} \quad v_2 \subset \{1, 2, \ldots, p\} \quad \text{et} \quad h_k(x_2) \neq 0 \quad \text{pour tout} \quad k \in v_2 \\ \text{et} \\ |u_1| + |v_1| < |u_2| + |v_2| \end{cases} \tag{3.6}$$

Cette notion de dominance permet de déterminer quelles solutions sont des optimums de Pareto.

3.2.3.2 Optimum de Pareto

L'optimum de Pareto est un concept utilisé en optimisation multi-objectifs pour définir les solutions qui ne peuvent pas être améliorées dans tous les objectifs sans détériorer au moins un autre objectif. Formellement, une solution x^* est considérée comme un optimum de Pareto si et seulement si pour tout x appartenant à S, il n'existe aucun autre point y dans S tel que $y \succ x$. L'ensemble de toutes les solutions de Pareto est appelé le "Pareto front".

3.2.3.3 Le Pareto front

Le concept de "Pareto front" représente toutes les solutions potentiellement optimales qui offrent un compromis entre les différents objectifs à optimiser. En d'autres termes, aucun point dans la Pareto front ne peut être amélioré dans tous les objectifs sans dégrader au moins un objectif. Mathématiquement, le Pareto front est défini comme suit :

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

La figure 3.2 présente des exemples de Pareto front pour des problèmes à deux et trois objectifs.

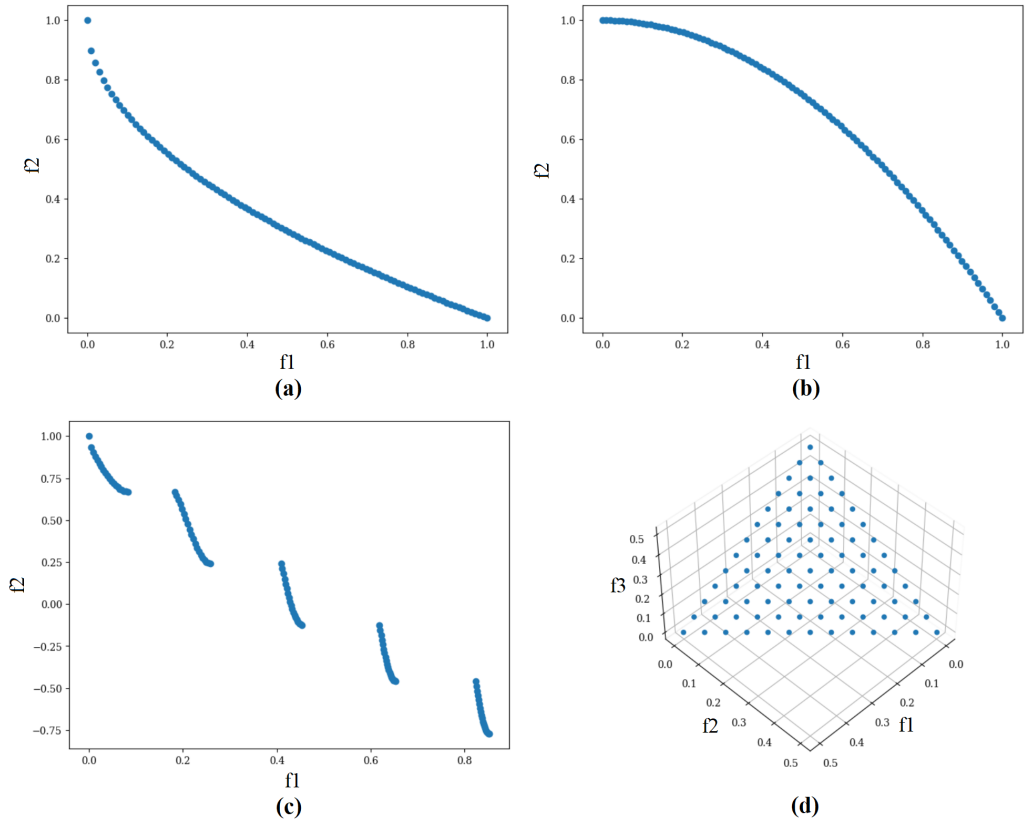


FIGURE 3.2 – Des exemples des différentes formes géométriques des Pareto front. Pour 2 objectifs : (a) convexe ; (b) concave ; (c) disjoint, pour 3 objectifs : (d) linéaire

3.2.3.4 Le point idéal

Le point idéal est un point dans l'espace des objectifs qui représente la meilleure valeur possible pour chaque objectif. Ce point idéal est souvent utilisé comme référence pour évaluer la performance des solutions d'un problème multi-objectifs.
Mathématiquement, le point idéal, est défini comme un vecteur $z^* = (z_1^*, z_2^*, ..., z_m^*)$ tel que :

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

La figure 3.3 représente une illustration du point idéal. Ce dernier n'appartient pas à l'espace des solutions réalisables car les objectifs sont, généralement, contradictoires.

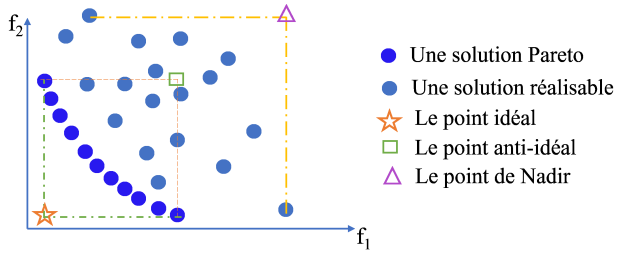


FIGURE 3.3 – Illustration du point idéal, point anti-idéal et point Nadir

3.2.3.5 Le point anti-idéal

Le point anti-idéal peut être défini comme les valeurs maximales de chacune des fonctions objectifs prises séparément pour toutes les solutions réalisables dans l'espace de recherche (figure 3.3). Il peut être défini comme suit :

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

où $f_i(x)$ représente la i-ème fonction objectif, et x est une solution réalisable dans l'espace de recherche.

3.2.3.6 Le point de Nadir

Le point de Nadir $N = (N_1, N_2, ..., N_m)$ est défini comme le vecteur des bornes supérieures de chaque objectif sur le Pareto front (Figure 3.3), tel que :

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

3.2.4 Les méthodes de résolution

Pour résoudre les problèmes d'optimisation multi-objectifs, plusieurs méthodes et techniques sont possibles. Ces méthodes visent à trouver un ensemble de solutions qui représentent le compromis optimal entre les différents objectifs.

3.2.4.1 Les types des méthodes de résolution

Il existe plusieurs types de méthodes de résolution pour les problèmes d'optimisation multi-objectifs, parmi lesquelles les plus courantes sont :

- Les méthodes exactes : sont des approches qui cherchent à trouver toutes les solutions optimales du problème en utilisant des techniques mathématiques formelles. Cependant, elles sont souvent confrontées à des défis de complexité computationnelle et limitations en termes de taille des problèmes. Bien qu'elles garantissent l'optimalité des solutions, elles deviennent rapidement inefficaces pour les problèmes de grande taille en raison de leur complexité exponentielle.
  - Les méthodes par énumération exhaustive : leur principe consiste à énumérer toutes les solutions possibles du Problème d'optimisation multi-objectifs (POM) et évalue chacune d'elles pour trouver la solution optimale. La méthode brute-force et la méthode de recherche exhaustive par arbre de décision sont deux exemples de ces approches.
  - La programmation Linéaire (PL) : leur principe consiste à formuler le POM sous forme de programme linéaire avec des objectifs multiples et des contraintes linéaires. Ensuite, des algorithmes tels que la méthode du Simplexe peuvent être utilisés pour résoudre le problème. Par exemple, l'algorithme Extended Multi-objective Simplex Algorithm (EMSA) [56] combine les principes du Simplexe pour résoudre le problème primal et le Dualité pour déterminer les directions d'optimisation pour chaque objectif, afin de trouver le front de Pareto optimal pour le problème d'optimisation multi-objectifs linéaires (MOLP).
  - La programmation Quadratique Mixte-Entière (PQME) : lorsque le POM inclut des termes quadratiques dans la fonction objectif ou dans les contraintes, il peut être formulé comme un problème de PQME. Des méthodes spécifiques aux PQME, telles les méthodes de résolution basées sur la décomposition, peuvent être utilisées pour résoudre ce type de problème. La méthode de "Benders Decomposition" [57] est un exemple de ces méthodes.
- Les méthodes de pondération : sont des techniques visant à résoudre les problèmes d'optimisation multi-objectifs en les transformant en des problèmes d'optimisation mono-objectif.

Elles consistent à attribuer des poids aux différents objectifs pour les agréger en une seule fonction objectif pondérée, généralement choisie pour refléter l'importance relative des objectifs. Une fois cette fonction pondérée définie, des méthodes d'optimisation mono-objectif standard sont appliquées pour trouver une solution. Bien que les méthodes de pondération offrent une approche simple, elles peuvent être sensibles au choix des poids et ne garantissent pas toujours des solutions efficaces sur le Pareto front.

— Les métaheuristiques : sont des approches qui reposent sur des règles empiriques et des techniques d'approximation pour trouver des solutions potentielles sans garantir l'optimalité. Contrairement aux méthodes exactes, ces méthodes ne garantissent pas la recherche exhaustive de toutes les solutions possibles, mais elles offrent souvent une alternative efficace pour explorer rapidement un grand espace de recherche et trouver des solutions de qualité acceptable.

— Les méthodes hybrides : visent à combiner les avantages de différentes approches pour résoudre les problèmes d'optimisation de manière plus efficace. Ces approches combinent souvent des techniques exactes ou des techniques ML avec des métaheuristiques ou d'autres méthodes pour améliorer l'efficacité et la performance de la résolution des problèmes d'optimisation en exploitant la complémentarité des différentes approches.

Nous nous intéressons dans la suite de cette section aux métaheuristiques et aux méthodes hybrides.

3.2.4.2 Les métaheuristiques

Les métaheuristiques sont des méthodes approchées qui cherchent à trouver des solutions proches de l'optimalité sans garantir la convergence vers une solution globale optimale. Ces approches sont souvent utilisées lorsque les problèmes d'optimisation sont complexes et que les méthodes exactes prennent trop de temps pour converger. Les méthodes approchées incluent une variété d'algorithmes. Cette section énumère les principales parmi ces approches.

(**a**) **Les métaheuristiques bio-inspirées** : Les métaheuristiques bio-inspirées tirent leur inspiration des comportements observés dans la nature pour résoudre des problèmes d'optimisation.

— **Un exemple : le GWO**

GWO [3] est un exemple des ces algorithmes, récemment proposé pour résoudre les problèmes d'optimisation mono-objectif. GWO est inspiré de la nature, basé sur le comportement social des loups gris pendant la chasse en groupe. Algorithme 1 décrit son fonctionnement pendant la résolution. En effet, il est conçu pour trouver la solution optimale à un problème d'optimisation en mettant à jour de manière itérative les positions des loups représentant les solutions qui constituent l'espace de recherche :

- les loups α, β et δ représentent les solutions optimales puisqu'ils ont une meilleure connaissance de la localisation de la proie. Pour cela, ils guident le processus de la chasse et dominent les loups ω ;
- le loup α est la meilleure solution ;
- les loups ω sont les autres solutions.

Algorithme 1 : Gray Wolf Optimizer (GWO)

Result : Meilleure solution trouvée
Initialization : Générer une population de loups gris avec des positions aléatoires
Initialiser les coefficients : a, A et C
Évaluer la fitness de chaque loup X_i dans la population
Identifier les loups X_α, X_β et X_δ en fonction de leurs valeurs de fitness
while *critère d'arrêt non atteint* **do**
  for *chaque loup dans la population* **do**
    Mettre à jour la position du loup :
    $\overrightarrow{D_\alpha} = |\overrightarrow{C_1}.\overrightarrow{X_\alpha} - \overrightarrow{X_i}(t)|, \overrightarrow{D_\beta} = |\overrightarrow{C_2}.\overrightarrow{X_\beta} - \overrightarrow{X_i}(t)|, \overrightarrow{D_\delta} = |\overrightarrow{C_3}.\overrightarrow{X_\delta} - \overrightarrow{X_i}(t)|$
    $\overrightarrow{X_{i1}} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.(\overrightarrow{D_\alpha}), \overrightarrow{X_{i2}} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.(\overrightarrow{D_\beta}), \overrightarrow{X_{i3}} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.(\overrightarrow{D_\delta})$
    $\overrightarrow{X_i}(t+1) = (\overrightarrow{X_{i1}} + \overrightarrow{X_{i2}} + \overrightarrow{X_{i3}})/3$
  Mettre à jour les coefficients
  Évaluer la fitness de chaque loup dans la population
  Identifier les loups X_α, X_β et X_δ en fonction des nouvelles valeurs de fitness
return X_α

— **Version améliorée de GWO : Improved Gray Wolf Optimizer (I-GWO)** Une version améliorée de GWO, appelée I-GWO, est proposée en 2021 dans [58]. Dans I-GWO, pour calculer et sélectionner $X_i(t+1)$, 2 positions candidates sont construites :

- La première position candidate, appelée $X_{i-GWO}(t+1)$, est calculée en utilisant la même méthode de calcul utilisée dans GWO.
- La deuxième position candidate, appelée $X_{i-DLH}(t+1)$, est calculée en utilisant une stratégie de recherche appelée Dimension learning-based hunting (DLH). Cette position est calculée à partir des coordonnées des voisins de $X_i(t)$ et un loup choisi parmi les éléments de l'espace de recherche aléatoirement.

Ensuite, une étape de sélection parmi les deux candidates est effectuée. La position choisie parmi $X_{i-GWO}(t+1)$ et $X_{i-DLH}(t+1)$ va représenter $X_i(t+1)$. X_i n'est modifiée que si le fitness de la nouvelle position est meilleur que celui de la position actuelle.

— **Version multi-objectifs de GWO : MOGWO**

Une version multi-objectifs de GWO est proposée en 2016 dans [2]. Cette version est appelée MOGWO. Deux notions sont ajoutées afin de traiter les optimisations multi-objectifs :

(i) La première notion est l'utilisation d'une archive qui consiste à un espace de stockage pour les solutions non-dominées. Elle a une capacité maximale admissible pour contenir les solutions. L'archive se compose de deux éléments principaux :

- Un contrôleur ayant comme rôle de contrôler l'ajout d'une nouvelle solution non dominée, NS, à l'archive :
  * Au début, l'archive est vide
  * Si l'archive est vide, la nouvelle solution NS est acceptée
  * S'il existe une solution dans l'archive qui domine NS, cette dernière est rejetée
  * S'il n'existe pas une solution dans l'archive qui domine NS, cette dernière est acceptée
  * Si NS et les éléments de l'archive ne se dominent pas, la nouvelle solution est acceptée
  * S'il y a des éléments dans l'archive qui sont dominées par la nouvelle solution ajoutée, ils doivent être supprimés.
- Une grille

  Les solutions dans l'archive, sont distribués et placées dans un espace de fonctions objectifs à m dimensions (rappelons que m est le nombre de fonctions objectif du problème). La position d'une solution dans l'espace de fonctions objectifs est déterminée selon ses valeurs objectifs.

  Une grille à m dimensions est utilisée pour suivre le degré d'encombrement dans les différentes régions de l'espace de fonctions objectifs. En effet, la grille est subdivisée en un nombre, à choisir, d'hypercubes (segments). Lorsqu'une solution est ajoutée à l'archive, sa position dans la grille (et donc l'hypercube auquel elle appartient) est calculée. Le rôle de la grille est de bien distribuer les solutions dans l'espace des fonctions objectifs et de maintenir leur diversification. En effet, si une nouvelle solution est ajoutée, elle est dirigée vers la partie de la grille avec le plus petit nombre d'éléments. Lorsque l'archive atteint sa capacité maximale, l'ajout d'une nouvelle solution se fait après la suppression d'un élément de l'archive. Dans ce cas, le mécanisme de la grille est déclenché pour ré-arranger la segmentation de l'espace et supprimer une solution du segment le plus encombré. Ensuite, la nouvelle solution est insérée dans le segment le moins encombré.

(ii) La deuxième notion est un mécanisme de sélection des meilleures solutions. En effet, dans l'algorithme GWO, les 3 meilleures solutions X_α, X_β et X_δ sont sélectionnées. Ensuite, ces solutions vont guider les agents pendant la recherche de l'optimum. Par contre, dans un problème multi-objectifs, la comparaison des solutions ne peut pas être réalisée d'une façon simple. Pour cela, MOGWO propose un mécanisme de sélection des solutions X_α, X_β et X_δ. Elles sont choisies parmi les éléments des segments les moins encombrés. Ce choix permet

d'orienter la recherche de solutions vers les zones non explorées. La méthode de sélection consiste à :

- Sélectionner le segment le moins encombré
- Si le nombre de solutions dans le segment sélectionné est $\geq$ à 3, trois solutions parmi elles sont choisies aléatoirement et attribuées à X_α, X_β et X_δ.
- Si le nombre de solutions dans le segment sélectionné est $<$ à 3, les autres solutions sont sélectionnées parmi les éléments du deuxième moins encombré segment.
- Si le deuxième moins encombré segment ne contient qu'une seule solution, le troisième moins encombré segment doit être trouvé pour attribuer un de ses éléments à X_δ.

Les métaheuristiques bio-inspirées sont des approches flexibles et robustes pour résoudre une grande variété de problèmes d'optimisation. Elles offrent une recherche efficace dans des espaces de solution vastes, mais leur performance peut varier selon le problème et nécessite souvent un ajustement minutieux des paramètres. Bien qu'elles ne garantissent pas toujours la convergence vers la solution optimale, ces méthodes restent largement utilisées pour leur capacité à trouver des solutions de qualité dans des délais raisonnables.

(**b**) **Les algorithmes génétiques** : Les algorithmes génétiques sont une classe de techniques d'optimisation inspirées du processus de sélection naturelle et de génétique.

La résolution d'un problème d'optimisation avec les algorithmes génétiques implique plusieurs étapes clés (Figure 3.4) :

— **Initialisation de la population** : une population initiale de solutions candidates est générée de manière aléatoire. Chaque solution x_i est représentée sous forme d'une séquence de gènes ou de valeurs ;

— **Évaluation du fitness** : Chaque solution de la population est évaluée en fonction de son fitness par rapport aux objectifs du problème d'optimisation. Cette évaluation est basée sur une fonction de fitness $f(x_i)$ qui quantifie la performance de chaque solution ;

— **Sélection des parents** : Les solutions les mieux adaptées sont sélectionnées pour devenir les parents de la prochaine génération. Plusieurs méthodes de sélection peuvent être utilisées, telles que la sélection par tournoi, la sélection par roulette, ou la sélection par rang ;

— **Reproduction** : Les parents sélectionnés se reproduisent pour produire une nouvelle génération de solutions. Cela implique souvent des opérations génétiques telles que la recombinaison (ou croisement) et la mutation. Le croisement combine les informations génétiques des parents pour créer des descendants, tandis que la mutation introduit de petites variations aléatoires dans les chromosomes pour maintenir la diversité génétique ;

— **Évaluation de la population** : La nouvelle population, composée des descendants et parfois de certains membres de la génération précédente, est évaluée en fonction de leur fitness ;

— **Remplacement des parents** : Les membres les moins performants de la population peuvent être éliminés pour faire de la place aux nouveaux descendants. Différentes stratégies de remplacement peuvent être utilisées, telles que le remplacement générationnel ou le remplacement par survie ;

— **Critère d'arrêt** : Le processus d'évolution se poursuit pendant un certain nombre d'itérations, ou jusqu'à ce qu'un critère d'arrêt prédéfini soit atteint. Ce critère peut être basé sur le nombre d'itérations, sur la convergence de la population vers un optimum, ou sur d'autres considérations spécifiques au problème.

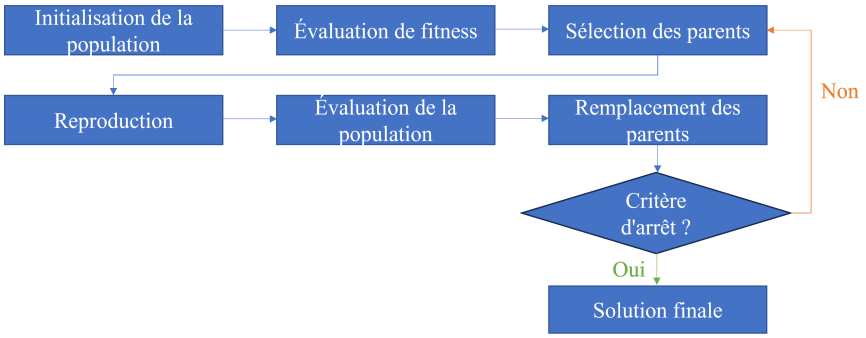


FIGURE 3.4 – Les étapes clés d'un algorithme génétique

Par exemple, le NSGA-III, proposé dans [59], est un algorithme génétique multi-objectifs qui utilise une méthode de classement améliorée pour classer les solutions en fonction de leur performance sur les objectifs multiples. Algorithme 2 décrit son principe. Il commence par l'initialisation d'une population de solutions aléatoires. Ensuite, ces solutions sont sélectionnées, croisées et mutées pour former une nouvelle population intermédiaire. Les objectifs de chaque solution dans cette population intermédiaire sont évalués. Ensuite, la méthode de classement NSGA-III est utilisée, prenant en compte les objectifs multiples et la similarité entre les solutions. Cette méthode implique une étape de non-dominance, où les solutions non dominées sont classées dans la première frontière de Pareto et les solutions dominées sont rejetées. Pour chaque frontière de Pareto, la densité de population est calculée en utilisant la distance de Crowding, puis les solutions sont triées en utilisant cette densité de population. Les solutions triées sont concaténées pour former une population classée. Enfin, une méthode de sélection de niche est employée pour choisir une population finale diversifiée et bien distribuée. Cette méthode consiste à diviser l'espace de recherche en niches, puis à sélectionner la solution la plus proche de la frontière de Pareto dans chaque niche. Si la taille de la population est inférieure à celle souhaitée, des solutions supplémentaires sont sélectionnées aléatoirement dans chaque niche. La population finale est ensuite triée en utilisant à nouveau les critères de non-dominance et de densité de population.

Algorithme 2 : NSGA-III

Result : Une population de solutions non-dominées bien distribuées
Initialisation : Créer une population initiale P_0 de taille N de solutions aléatoires
while *critère d'arrêt non atteint* **do**
  Sélection : Sélectionner N solutions de P_{t-1} pour former une population temporaire R_t
  Crossover : Appliquer l'opérateur de crossover à R_t pour générer une population intermédiaire Q_t
  Mutation : Appliquer l'opérateur de mutation à Q_t pour générer une population intermédiaire Q'_t
  Évaluation : Évaluer les objectifs de toutes les solutions dans Q'_t
  Classement : Classer les solutions dans $P_{t-1} \cup Q'_t$ en utilisant la méthode de classement NSGA-III
  Niche : Sélectionner N solutions de $P_{t-1} \cup Q'_t$ en utilisant la méthode de sélection de niche pour former la population P_t
return P_t

Les méthodes génétiques offrent une exploration efficace de l'espace de recherche et peuvent gérer plusieurs solutions simultanément. Elles sont simples à mettre en oeuvre mais peuvent converger lentement, être sensibles aux paramètres et avoir du mal avec des problèmes à grande échelle.

(**c**) **Les essaims particulaires** Les méthodes des Essaims Particulaires sont des algorithmes d'optimisation basés sur des modèles inspirés du comportement social des animaux en groupe et de leurs interactions auto-organisées, tels que les essaims d'oiseaux ou les bancs de poissons. Ces méthodes visent à trouver la meilleure solution à un problème donné en itérant sur un ensemble de particules, chacune représentant une solution potentielle. Chaque particule est caractérisée par une position dans l'espace de recherche et une vitesse associée.

— **PSO** [60] est un exemple de ces algorithmes qui permet de résoudre les problèmes d'optimisation mono-objectifs. Il est basé sur la population, inspiré du comportement social du vol d'oiseaux. Algorithme 3 représente le principe de PSO. Il cherche à résoudre un problème en itérant sur un ensemble de particules, chaque particule représentant une solution potentielle avec une position et une vitesse dans l'espace de recherche. Les positions et vitesses des particules sont initialement aléatoires. À chaque itération, chaque particule met à jour sa position et sa vitesse en fonction de sa meilleure position personnelle et de la meilleure position globale parmi toutes les particules, en utilisant des facteurs d'inertie et d'accélération, ainsi que des valeurs aléatoires. L'algorithme continue jusqu'à ce qu'un critère d'arrêt, tel qu'un nombre maximal d'itérations, soit atteint, puis retourne la meilleure position globale trouvée, représentant la solution optimale. Dans cet algorithme, $x_{i,d}(t)$ représente la position de la particule i dans la dimension d à l'itération t, et $v_{i,d}(t)$ est la vitesse correspondante. La meilleure position personnelle de la particule i dans la dimension d est désignée par $p_{i,d}$, et $p_{g,d}$ représente la meilleure position globale parmi toutes les particules dans la dimension d.

ω est le poids d'inertie, c_1 et c_2 sont des constantes d'accélération, et r_1 et r_2 sont des valeurs aléatoires distribuées uniformément dans [0, 1].

Algorithme 3 : Particle Swarm Optimization (PSO)

Result : Solution optimale
Initialisation : Initialiser les particules avec des positions et des vitesses aléatoires
while *critères d'arrêt non atteints* **do**
  for *chaque particule* i **do**
    Mettre à jour la meilleure position personnelle : $p_i \leftarrow$ meilleure position de la particule i
    Mettre à jour la meilleure position globale : $p_g \leftarrow$ meilleure position parmi toutes les particules
    for *chaque dimension* d **do**
      Mettre à jour la vitesse de la particule : $v_{i,d}(t+1) \leftarrow \omega \cdot v_{i,d}(t) + c_1 \cdot r_1 \cdot (p_{i,d} - x_{i,d}(t)) + c_2 \cdot r_2 \cdot (p_{g,d} - x_{i,d}(t))$
      Mettre à jour la position de la particule : $x_{i,d}(t+1) \leftarrow x_{i,d}(t) + v_{i,d}(t+1)$
    Évaluer la nouvelle position
return *Meilleure position globale* p_g

L'algorithme PSO est connu pour sa simplicité, sa facilité de mise en oeuvre et sa convergence rapide. Il a été appliqué à divers problèmes d'optimisation, y compris l'optimisation continue, discrète et combinatoire. Cependant, il peut souffrir de convergence anticipée et d'immobilisme, surtout pour les problèmes d'optimisation de grande dimension.

Pour résoudre ces problèmes, diverses modifications et extensions de l'algorithme PSO ont été proposées, telles que :

- Le Bare Bones Particle Swarm Optimization (BBPSO) [61] : est une modification de PSO qui vise à simplifier les équations et à réduire le nombre de paramètres. L'objectif est de trouver un équilibre entre l'exploration et l'exploitation en ajustant dynamiquement le comportement de recherche des particules.
- Le Quantum-behaved Particle Swarm Optimization (QPSO) [62] : est une variante de PSO qui utilise une approche inspirée de la physique quantique pour mettre à jour les positions et les vitesses des particules, ce qui conduit à de meilleures propriétés de convergence et de robustesse.

— **Multi-objective Particle Swarm Optimization (MOPSO)** : MOPSO [60] est une modification de PSO conçue pour gérer les problèmes d'optimisation multi-objectifs. En effet, il introduit une représentation des solutions prenant en compte plusieurs objectifs simultanément, utilise une fonction de domination pour évaluer la qualité des solutions par rapport à ces objectifs, maintient une archive des solutions non dominées pour former le front de Pareto, et met à jour la meilleure position globale en considérant cette archive. Ces ajouts permettent à MOPSO de gérer efficacement les problèmes multi-objectifs en explorant un ensemble diversifié de solutions optimales. Algorithme 4 représente les étapes de MOPSO.

Algorithme 4 : Multi-objective Particle Swarm Optimization (MOPSO)

Result : Ensemble de solutions non-dominées P
Initialisation : Générer une population de N particules avec des positions aléatoires $\mathbf{x}_i$ et des vitesses $\mathbf{v}_i$
for $i = 1$ *à* N **do**
  Évaluer la performance de la particule i en utilisant les fonctions objectif $f_m(\mathbf{x}_i)$, où $m = 1, 2, \ldots, M$
  if $f_m(\mathbf{x}_i) < f_m(\mathbf{p}_i)$ *pour tout* m **then**
    $\mathbf{p}_i = \mathbf{x}_i$; $F(\mathbf{p}_i) = F(\mathbf{x}_i)$
Mettre à jour les meilleures positions globales $\mathbf{g}_m$ et la performance $F(\mathbf{g}_m)$ de l'essaim en se basant sur les valeurs de performance de toutes les particules
while *critère d'arrêt non atteint* **do**
  for $i = 1$ *à* N **do**
    Mettre à jour la vitesse et la position de la particule i en utilisant les équations :
    $\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1 r_1(\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2(\mathbf{g}_m(t) - \mathbf{x}_i(t))$
    $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1)$
    Évaluer la performance de la nouvelle position de la particule i en utilisant les fonctions objectif $f_m(\mathbf{x}_i)$, où $m = 1, 2, \ldots, M$
    if $f_m(\mathbf{x}_i) < f_m(\mathbf{p}_i)$ *pour tout* m **then**
      $\mathbf{p}_i = \mathbf{x}_i$
      $F(\mathbf{p}_i) = F(\mathbf{x}_i)$
    if $f_m(\mathbf{x}_i) < f_m(\mathbf{g}_m)$ *pour tout* m **then**
      $\mathbf{g}_m = \mathbf{x}_i$
      $F(\mathbf{g}_m) = F(\mathbf{x}_i)$
  Mettre à jour l'ensemble de solutions non-dominées P en utilisant les règles suivantes :
  if *la particule* i *n'est dominée par aucune autre particule dans* P **then**
    Ajouter la particule i à P
  else if *la particule* i *est dominée par une autre particule dans* P **then**
    Retirer la particule i de P
  else if *la particule* i *est dans* P *et est dominée par une autre particule dans* P **then**
    Retirer la particule i de P
return P

Dual Multi-Objective Particle Swarm Optimization (DMOPSO) est une version améliorée de MOPSO, proposée par [63]. Elle inclut des objectifs supplémentaires visant à optimiser à la fois la préférence de l'utilisateur et la diversité.

Les méthodes des Essaims Particulaires (PSO) offrent une approche simple et accessible pour résoudre divers problèmes d'optimisation, grâce à leur facilité de mise en oeuvre et à leur efficacité dans l'exploration de vastes espaces de recherche. Elles peuvent rapidement converger vers des solutions de qualité, mais peuvent également être sensibles aux paramètres et avoir tendance à converger vers des solutions optimales locales dans des espaces de recherche complexes.

(**d**) **La programmation évolutionnaire (PE)** : Les algorithmes de la programmation évolutionnaire imitent le processus de sélection naturelle et de reproduction au sein d'une population de solutions candidates. Par exemples, le Electrostatic Potential Energy Evolutionary Algorithm (ESPEA) [64] et le Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [65] sont

parmi ces algorithmes. ESPEA utilise le concept d'énergie potentielle électrostatique pour estimer les solutions obtenues au (PF_{true}). MOEA/D est une méthode largement utilisée pour résoudre des problèmes d'optimisation multi-objectifs. Il divise un problème multi-objectifs en plusieurs sous-problèmes mono-objectif, puis les optimise simultanément. Cette approche consiste à créer un ensemble de vecteurs de poids qui couvrent uniformément l'espace des objectifs, et à optimiser ensuite les sous-problèmes associés à ces vecteurs de poids. Les solutions de ces sous-problèmes sont ensuite combinées pour former une population variée et bien convergée qui représente approximativement le Pareto front.

Le MOEA/DD [66] estune version améliorée de MOEA/D. Algorithme 5 décrit la résolution d'un POM par MOEA/DD. Il initialise la population parente P, le jeu de vecteurs de poids W, et l'ensemble d'index de voisinage E. Ensuite, tant que le critère d'arrêt n'est pas satisfait, il sélectionne des parents de reproduction dans toute la population avec une faible probabilité $1-\lambda$, où $\lambda \in [0,1]$. Il applique ensuite le croisement binaire simulé (Simulated Binary Crossover (SBX)) et la mutation polynomiale pour obtenir un ensemble S de solutions variées. Pour chaque solution dans S, il met à jour la population parente P. Enfin, il retourne la population mise à jour P.

Algorithme 5 : MOEA/DD

Result : Population P

Initialisation :

- Initialiser P : la population parente
- Initialiser W : l'ensemble des vecteurs de poids
- Initialiser E : l'ensemble des index de voisinage

while *le critère d'arrêt n'est pas rempli* **do**
  for $i = 1, 2, \ldots, N$ **do**
    Sélection de reproduction : sélectionner les parents reproducteurs dans toute la population avec une faible probabilité $1-\lambda$, où $\lambda \in [0,1]$
    Variation : appliquer le croisement binaire simulé (SBX) [67] et la mutation polynomiale [68] pour obtenir S
    foreach $x \in S$ **do**
      $P \leftarrow$ mise_à_jour_population(P, x)

return *P*

Diverses variantes modifiées de MOEA/D ont été développées, notamment 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] et MOEA/DCDP [71].

La programmation évolutionnaire offre une approche robuste et flexible pour résoudre une variété de problèmes d'optimisation. Elle permet de trouver des solutions robustes pour des problèmes complexes et dynamiques, ainsi que dans la résolution de problèmes multi-objectifs et de contraintes non linéaires. La PE peut être moins sensible aux paramètres et plus adaptable que cer-

taines autres techniques d'optimisation. Cependant, elle peut être confrontée à des défis tels que la lenteur de convergence, le besoin d'évaluations intensives de la fonction objectif et la difficulté de configuration des paramètres.

(**e**) **Le recuit simulé** : Les algorithmes de type recuit simulé sont une classe d'algorithmes d'optimisation stochastiques inspirés du processus de refroidissement et de cristallisation d'un matériau fondu pour atteindre un état énergétique minimal, ce qui est analogique à la recherche de la solution optimale dans un espace de recherche. L'algorithme commence par une solution initiale et explore l'espace des solutions en acceptant ou en rejetant des solutions voisines selon une probabilité déterminée par une fonction de température. Au fil du temps, la température diminue, ce qui réduit la probabilité d'accepter des solutions pires que la solution actuelle, permettant à l'algorithme de converger vers une solution optimale ou proche de l'optimalité. Le Simulated Annealing (SA) [72] est à l'origine de ces algorithmes. Il est conçu pour résoudre les problèmes mono-objectifs. Algorithme 6 décrit son principe. Il débute avec une solution initiale aléatoire S et une température initiale T. À chaque itération, il génère une solution voisine S' et évalue sa qualité par rapport à la solution actuelle. La décision d'accepter ou de rejeter la solution voisine dépend de la différence de coût ΔE et de la température courante T. La température est mise à jour selon un schéma de refroidissement à chaque itération. L'algorithme s'arrête une fois que la température atteint sa valeur minimale prédéfinie T_{min}, et il retourne la meilleure solution trouvée.

Algorithme 6 : Recuit simulé (Simulated Annealing)

Result : Meilleure solution trouvée
Initialisation : Générer une solution initiale aléatoire S
Initialiser la température T à une valeur initiale
while $T > T_{min}$ **do**
    Générer une solution voisine S' à partir de S
    Calculer la différence de coût $\Delta E = E(S') - E(S)$
    if $\Delta E < 0$ **then**
        Accepter la solution voisine S'
    else
        Générer un nombre aléatoire p entre 0 et 1
        Calculer la probabilité d'acceptation $P = e^{-\Delta E/T}$
        if $p < P$ **then**
            Accepter la solution voisine S'
        else
            Rejeter la solution voisine S'
    Mettre à jour la température T selon le programme de refroidissement
return S

[72] propose une version améliorée de SA. Ce nouvel algorithme est désigné sous le nom de "Simulated Annealing de l'Orikaeshi Tanren" (Orikaeshi Tanren Simulated Annealing (OTSA)). La méthode proposée exploite le concept de pliage et de réchauffement pour concevoir deux nouveaux opérateurs. L'opérateur de pliage réduit progressivement la taille de l'espace de recherche,

accentuant ainsi l'exploitation grâce à la concentration des agents de recherche. Parallèlement, le processus de réchauffement relance le refroidissement une fois le critère d'arrêt atteint, permettant ainsi des mouvements significatifs des agents de recherche pour éviter les solutions optimales locales.

Les algorithmes de recuit simulé offrent une approche simple et efficace pour résoudre des problèmes complexes en évitant les solutions optimales locales grâce à la stratégie de température décroissante. Ils ne nécessitent pas de connaissances préalables sur la structure du problème et peuvent fournir des solutions de qualité. Cependant, leur performance peut dépendre de l'ajustement minutieux des paramètres et être influencée par la dimensionnalité et la complexité du problème, sans garantie de convergence vers la solution optimale.

(f) La recherche tabou (RT) : La méthode de recherche tabou (Tabu Search (TS)) [73] est une méthode de recherche locale approchée (métaheuristique) qui peut être utilisée pour résoudre les POMs. Elle est basée sur l'idée d'explorer l'espace de recherche en générant de nouvelles solutions à partir de solutions existantes, et les solutions qui ont déjà été visitées sont stockées dans une liste de mémoire tabou, qui est utilisée pour empêcher la recherche de revenir à des solutions déjà visitées. Algorithme 7 contient les étapes de résolution d'un MOP en appliquant la méthode de recherche tabou. Il commence par initialiser une solution (s^*) de manière aléatoire et crée une mémoire tabou (POP) initialement vide. La fonction Evaluate() évalue la valeur de fitness de la solution initiale. Ensuite, l'algorithme maintient une solution optimale (S) et un compteur d'itérations (it). À chaque itération, il recherche un ensemble de solutions voisines avec la fonction findNeighborSet(), trouve la meilleure solution dans cet ensemble avec la fonction findBest(), met à jour la mémoire tabou (avec la fonction update()), et actualise la solution optimale S. Ce processus se répète jusqu'à ce que le nombre maximal d'itérations, défini par max_it, soit atteint. L'algorithme vise à explorer l'espace des solutions tout en évitant les mouvements interdits mémorisés dans la mémoire tabou pour converger vers une solution optimale.

Les avantages de la recherche tabou comprennent une exploration efficace de l'espace des solutions, une capacité à échapper aux solutions optimales locales et une adaptation aux problèmes complexes. Cependant, elle peut être sensible aux paramètres, complexe à mettre en oeuvre et dépendante des conditions initiales.

3.2.4.3 Les méthodes hybrides

Les méthodes hybrides souvent associent différentes métaheuristiques entre elles, ou combinent des techniques exactes ou des méthodes ML avec des métaheuristiques, dans le but d'améliorer l'efficacité et les performances de résolution des problèmes d'optimisation en exploitant les avantages complémentaires des différentes approches. Cette sous-section propose deux exemples d'approches d'hybridation.

Algorithme 7 : Algorithme de recherche tabou pour les problèmes d'optimisation multi-objectifs

Result : Solution optimale S
$s^* \leftarrow$ InitialiserSolution()
Initialiser la mémoire tabou
$POP =$ null
Évaluer(s^*)
$S \leftarrow s^*$
$it \leftarrow 0$
while $it \leq max\_it$ **do**
  $it \leftarrow it + 1$
  $nbs \leftarrow$ TrouverEnsembleVoisins(s^*)
  $b \leftarrow$ TrouverMeilleur(nbs, it)
  mettreÀJour(POP, it)
  $S \leftarrow b$

— **Exemple 1, algorithme génétique + recherche tabou** : [74] propose un algorithme hybride qui combine un algorithme génétique avec une recherche tabou pour résoudre le Flow Shop Scheduling Problem (FSSP). L'objectif principal est de minimiser le makespan, c'est-à-dire le temps total nécessaire pour achever tous les travaux dans l'atelier. L'organigramme de l'algorithme hybride est présenté par la figure 3.5. Il débute en générant une population initiale de solutions via l'algorithme génétique. Par la suite, il parcourt la population et applique l'algorithme de recherche tabou à chaque solution. En cas de découverte d'une solution améliorée, elle est intégrée à la population et la liste tabou est actualisée. L'algorithme se termine dès qu'une solution avec une valeur de fitness inférieure à un seuil prédéfini est détectée. Les auteurs fournissent également des résultats expérimentaux afin d'évaluer l'efficacité de leur algorithme hybride. Ils comparent les performances de cet algorithme avec celles d'autres méthodes telles que un algorithme génétique, l'algorithme de recherche tabou et un algorithme de recherche locale. Les résultats indiquent que l'algorithme hybride surpasse les autres méthodes en termes de recherche de la solution optimale et de minimisation du makespan, ce qui met en évidence la contribution de la combinaison de deux techniques. Cependant, l'algorithme a quelques limitations, notamment son utilisation de deux algorithmes métaheuristiques, ce qui signifie que l'algorithme peut ne pas toujours trouver la solution optimale, surtout pour les problèmes complexes, sa spécificité au problème flow shop, son coût computationnel potentiellement élevé et sa difficulté à être parallélisable.

— **Exemple 2, essaims particulaires + Machine Learning** : [75] propose un algorithme hybride pour résoudre le problème complexe de la planification et de l'allocation des ressources dans les environnements de fog computing. L'algorithme Enhanced Multi-Objective Particle Swarm Optimization with Clustering (EMOPSOC) combine le MOPSO avec des techniques de Machine Learning. L'objectif est d'optimiser la planification des noeuds de fog en mi-

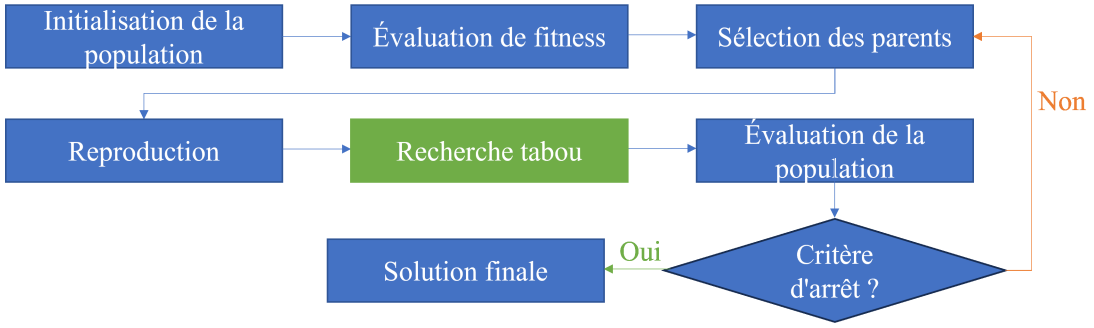


FIGURE 3.5 – Organigramme de l'algorithme hybride : algorithme génétique + recherche tabou

nimisant la latence entre les dispositifs IoT et les noeuds de fog, en réduisant le gaspillage de ressources, en équilibrant les charges de travail et en améliorant l'utilisation efficace des ressources. La figure 3.6 présente l'organigramme de EMOPSOC. L'algorithme débute par l'initialisation d'une population de particules avec des positions et des vitesses aléatoires, suivie de l'évaluation de la fitness pour chaque particule. Ensuite, les meilleures positions personnelles et leurs fitness sont mises à jour. La technique ML de clustering est ensuite appliquée pour regrouper les particules en clusters en fonction de leurs similitudes. Dans chaque cluster, la meilleure particule globale est sélectionnée pour chaque particule. Les vitesses et les positions de chaque particule sont mises à jour en utilisant les meilleures positions personnelles et globales. Le processus continue jusqu'à ce qu'un critère d'arrêt soit atteint, comme un nombre maximal d'itérations ou une convergence. Les simulations effectuées ont montré une amélioration significative, par l'algorithme proposé, des performances en termes de recherche d'un ensemble de solutions non dominées et de minimisation du coût total, ainsi que de maximisation de l'utilité totale du problème de planification des ressources et de placement des applications. Cependant, EMOPSOC présente des inconvénients, notamment une complexité computationnelle accrue due à l'étape de clustering, une difficulté dans le choix de la méthode de clustering adaptée, un risque de convergence prématurée vers des solutions sous-optimales et une exploration limitée de l'espace de recherche, rendant difficile la prédiction de ses performances sur différents problèmes.

La combinaison des métaheuristiques avec des méthodes exactes et des techniques ML offre plusieurs avantages, notamment une précision améliorée des solutions, une exploration et une exploitation plus efficaces de l'espace de recherche, une meilleure scalabilité, ainsi qu'une flexibilité et une adaptabilité accrues aux exigences changeantes. Cependant, cette approche présente également des inconvénients, notamment une complexité accrue des algorithmes, un coût computationnel plus élevé et des défis dans le choix des méthodes appropriées.

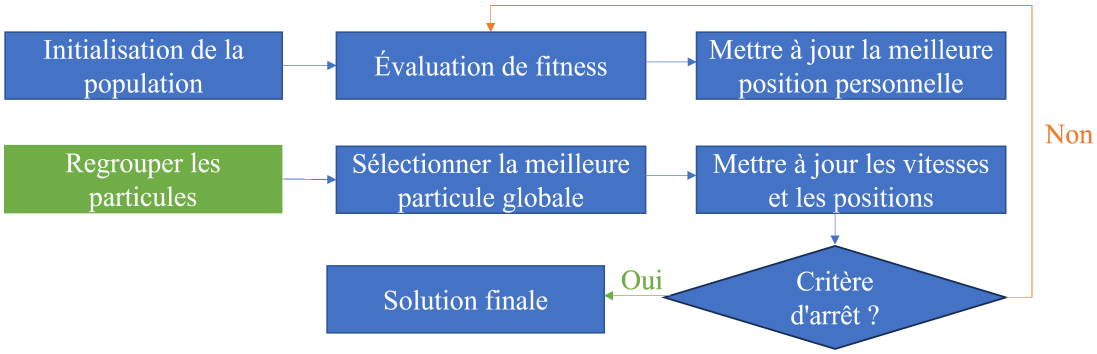


FIGURE 3.6 – Organigramme de l'algorithme hybride EMOPSOC : MOPSO + technique ML de clustering

3.2.5 Les critères de performance des méthodes de résolution

Les indicateurs de performance sont des outils utilisés pour évaluer la capacité de convergence de PF, l'ensemble de solutions non dominées obtenues par la méthode de résolution, vers le vrai Pareto Front, dénommé (PF_{true}). Ils permettent également d'analyser la distribution des solutions obtenues dans l'espace des objectifs. Les indicateurs les plus courants sont les suivants [76] :

— **"Generational Distance" (GD)** : cette mesure évalue la proximité des solutions de PF par rapport à PF_{true} en calculant la distance euclidienne entre chaque solution non dominée et sa solution la plus proche dans PF_{true}. Une valeur de "Generational Distance" plus faible indique une meilleure convergence. Le "Generational Distance" peut être calculé conformément à l'équation 3.11.

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

Généralement, p=2.

— **Inverted Generational Distance (IGD)** : dérivée de "Generational Distance", cette mesure évalue à la fois la convergence et la distribution de PF. Une valeur plus faible de l'IGD indique une meilleure performance globale de l'algorithme. Cet indicateur peut être calculé selon l'équation 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)** : cette mesure quantifie le volume de l'espace objectif dominé par les solutions de PF. Elle offre une évaluation combinée de la convergence et de la distribution des solutions. Une valeur plus élevée de HV correspond à une meilleure performance. Une représentation courante de l'équation pour HV est la suivante 3.13 :

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

Où :

- m est le nombre d'objectifs et $\mathbb{R}^M$ représente l'espace objectif m-dimensionnel ;
- Q représente le sous-ensemble de l'espace objectif en cours de considération ;
- $Volume(\cdot)$ représente la fonction de volume.

Cette intégrale est généralement approximée à l'aide de diverses techniques en raison de la complexité computationnelle du calcul de l'hypervolume exact. Une approche courante consiste à discrétiser l'espace objectif en petits hypercubes et à calculer le volume en fonction du nombre d'hypercubes dominés par le Pareto Front.

— **Walking Fish Group Hypervolume (WFGHV)** : c'est une version améliorée de HV. Il est calculé en utilisant l'algorithme Walking Fish Group (WFG) introduit dans [77]. Cette méthode est conçue pour permettre un calcul rapide de l'hypervolume, même pour des problèmes comportant 5 objectifs ou plus. Le WFGHV peut être calculé selon l'équation :

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

Où N est le nombre de solutions dans PF, m est le nombre d'objectifs, V_i est le volume de l'espace objectif dominé par la i-ème solution dans PF, et l'intégrale est prise sur l'ensemble de l'espace objectif excluant l'union des volumes dominés de toutes les solutions dans PF.

— **Normalized Hypervolume (NHV)** : calculé lorsque PF_{true} est connu, il peut être calculé à partir de HV ou WFGHV (équation 3.15). Les valeurs de l'indicateur NHV se situent entre 0 (meilleure valeur) et 1 (pire valeur).

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

— **Maximum Spread (MS)** : cet indicateur reflète la dispersion des solutions. Une valeur de MS plus élevée indique une distribution plus étendue de PF. Il peut êttre calculé comme suit :

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

Où m est le nombre d'objectifs, y_{ik} et y_{jk} représentent respectivement la k^{me} valeur objective des i^{me} et j^{me} solutions dans PF.

— **Spacing Metric (SP)** : elle mesure la variance de distance entre les solutions voisines dans le PF trouvé. Une valeur plus faible de SP correspond à une meilleure répartition des solutions. Équation 3.17 indique le calcul de SP :

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

Où : N est le nombre de solutions dans le PF en cours d'évaluation et d_i est la distance entre la i^{me} solution et son voisin le plus proche sur le PF.

— **Diversity (D)** : elle évalue la dispersion des solutions non dominées dans le PF. Une valeur plus élevée de D indique une meilleure performance globale de l'algorithme. La diversité est souvent évaluée à l'aide de diverses mesures ou calculs adaptés au problème spécifique et aux objectifs. Par exemple, une façon courante de mesurer la diversité consiste à calculer la distance moyenne entre les solutions dans le PF. Cependant, l'équation exacte pour ce calcul dépendrait du contexte spécifique du problème d'optimisation et de la métrique choisie pour évaluer la diversité.

3.2.6 Comparaison et revue des 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 tableau 3.1 expose une étude sur des travaux récents proposant des algorithmes d'optimisation.

TABLEAU 3.1: Comparaison et revue des méthodes récentes de résolution des problèmes d'optimisation

Papier	Année	Type	Nature	Approche/Objectifs	Techniques	Problème(s) d'évaluation	Nombre d'objectifs	Résultats	Inconvénients
[75]	2023	hybride	Multi-objectifs	EMOPSOC	MOPSO et des techniques de Machine Learning pour le clustering	La planification et l'allocation des ressources dans les environnements de fog computing	3	IGD, Hypervolume : 80% increase	algorithme spécifique à un type particulier de problèmes ; une complexité computationnelle accrue ; une difficulté dans le choix de la méthode de clustering adaptée
[78]	2023	Métaheuristique	Multi-objectifs	Un nouvel algorithme pour réaliser des optimisations multi-objectifs de conceptions de bâtiments réels	Une approche probabiliste bayésienne	Problème BNH [79] ; Conception "Two-Bar truss" [80] ; "Gear Train Design" [81]	2	Fraction des solutions réelles du PF trouvée : 31.58% ; Pourcentage de mauvaises solutions optimales identifiées par l'algorithme : 8.75%	L'algorithme nécessite un ajustement des paramètres pour des performances optimales ; L'applicabilité à d'autres types de problèmes nécessite d'être vérifiée ; Coût computationnel élevé
[82]	2022	Essaims particulaires	Mono-objectifs	QSPSO : combiner les principes de QPSO et le concept de solitons	solitons	des fonctions de test de référence (F1-F21)	1	nombre de réussites : 16-20 et temps de calcul : 0.81s-3.25s	L'algorithme utilise plusieurs paramètres qui doivent être réglés ; un coût computationnel plus élevé en raison des calculs supplémentaires pour les composants de la mécanique quantique et des solitons
[83]	2022	Hybride	Multi-objectifs	Ant-Colony Non-dominated Sorting Genetic Algorithm III (AcNSGA-III) : un nouvel algorithme hybride pour résoudre le problème du routage dans un réseau IoT en intérieur	NSGA-III et ACO [84]	problèmes DTLZ et le routage dans un réseau IoT	6	IGD : 0.115-9.624 ; HV : 0.901-0.982	Applicabilité à d'autres types de réseaux nécessiterait des investigations supplémentaires ; Dépendance aux paramètres spécifiques et leur ajustement pour une performance optimale
[58]	2021	Bio-inspiré	Mono-objectifs	I-GWO : une version améliorée de GWO	DLH	"Pressure vessel problem", "Welded beam problem" et "IEEE 30-bus test system"	1	Efficacité globale : 94.2%	La complexité supplémentaire par rapport à GWO peut rendre l'algorithme plus difficile à mettre en oeuvre ; le choix de bons paramètres peut être difficile et nécessite des essais et des erreurs

Suite à la page suivante

Tableau 3.1 – suite de la page précédente

Papier	Année	Type	Nature	Approche/Objectifs	Techniques	Problème(s) d'évaluation	Nombre d'objectifs	Résultats	Inconvénients
[85]	2021	Métaheuristique	Mono-objectifs	AoA : inspiré par le comportement de distribution des opérateurs arithmétiques tels que l'addition, la multiplication, la division, etc., dans la résolution des problèmes arithmétiques.	des opérateurs arithmétiques	des fonctions de test de référence (F1-F13), 5 problèmes de conception en ingénierie	1	rang selon valeur de fitness moyenne obtenue : 1 pour 12 fonctions	Coût computationnel élevé en raison des calculs supplémentaires nécessaires pour les opérations arithmétiques ; AoA a été testé sur un ensemble restreint de problèmes de référence comportant un petit nombre de variables.
[74]	2021	hybride	Mono-objectifs	GA-TS : algorithme hybride pour résoudre le problème d'ordonnancement de type "flow shop"	algorithme génétique et recherche tabou	FSSP	1	Erreur moyenne : 3.05% ; pourcentage d'augmentation du makespan : 14.66%	L'algorithme est spécifiquement conçu pour résoudre le FSSP ; Il a été testé sur un ensemble limité d'instances du FSSP ; Un coût computationnel élevé en raison de combinaison de deux métaheuristiques ;
[86]	2021	Évolutionnaire	Multi-objectifs	MaOEADRA : un algorithme basé sur la dominance et la décomposition avec adaptation du point de référence pour sélectionner les solutions pour la prochaine génération	une stratégie d'adaptation du point de référence	des fonctions de test de référence : DTLZ, WFG [4], IDTLZ [87], MaF [88], MLDMP [89] et MPDMP [90]	jusqu'à 15	HV : le meilleur dans 81,37% des cas ; IGD : le meilleur dans 84,39% des cas	Les performances ne sont pas évaluées sur un problème autre que les problèmes de référence ; Le choix des points de référence et de la stratégie de décomposition peut influencer les performances ; Coût computationnel élevé
[57]	2019	PQME	Mono-objectifs	Un algorithme exact pour résoudre de manière efficace les Partial Set Covering Location Problem (PSCLP) et Maximal Covering Location Problem (MCLP) réalistes.	branch-and-Benders-cut	MCLP et PSCLP	1	Temps d'exécution moyen : 0.35s pour PSCLP (100000 instances) et 26.08s pour MCLP (100000 instances)	La méthode nécessite une bonne solution initiale, ce qui peut être difficile à obtenir pour des problèmes à grande échelle ; Les performances nécessitent d'être vérifiées sur d'autres problèmes d'optimisation
[72]	2019	Recuit simulé	Mono-objectifs	OTSA : une version améliorée de SA inspirée par les techniques de métallurgie anciennes	Orikaeshi Tanren	des fonctions de test de référence, des problèmes d'optimisation en ingénierie	1	pourcentage des meilleures solutions générées par acOTSA : 75%, temps d'exécution moyen : 0.78s	Un ajustement de plusieurs paramètres est nécessaire, ce qui peut être long et coûteux en termes de calcul ; la complexité de l'algorithme

Synhèse L'étude des 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, avec de bonnes performances 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.

3.3 Les techniques Deep Learning

Le Machine Learning, une discipline clé de l'intelligence artificielle, repose sur des algorithmes qui permettent aux systèmes informatiques de détecter des motifs et des structures complexes dans les données, ce qui leur permet de prendre des décisions, de faire des prédictions et d'améliorer leurs performances au fil du temps. Parmi les techniques ML, le DL est une technique avancée de premier plan qui s'appuie sur des réseaux de neurones artificiels composés de multiples couches. Contrairement aux méthodes de ML traditionnelles qui nécessitent souvent une ingénierie manuelle des caractéristiques, le DL est capable d'apprendre automatiquement à partir des données brutes, ce qui le rend extrêmement puissant pour la modélisation de données complexes et non structurées. Pour cela, les techniques de DL peuvent être appliquées à une large gamme d'applications.

3.3.1 Principaux domaines d'applications du Deep Learning

Les techniques DL offre la possibilité de résoudre une grande variété de problèmes dans différents domaines, notamment :

— La régression : prédiction d'une ou plusieurs variables continues en fonction d'autres variables d'entrée ;

— La classification : consiste à attribuer des étiquettes ou des catégories à des données en fonction de leurs caractéristiques ;

— Segmentation d'images : partitionnement d'une image en plusieurs segments ou régions pour l'analyse et le traitement ultérieurs ;

— Détection d'objets : identification et localisation d'objets spécifiques dans une image ou une vidéo.

— Reconnaissance d'activités : identification et classification d'activités spécifiques à partir de séquences vidéo ;

— Génération de contenu : création de contenu automatique, tels que la génération de texte, d'images ou de musique.

— Traduction : traduction automatique entre différents modes de communication, tels que la traduction de texte en langage des signes ;

— Analyse de graphes : extraction de motifs et d'informations importantes à partir de données de graphes, telles que les réseaux sociaux ou les réseaux de transport ;

— Sélection automatique de caractéristiques : identification automatique des caractéristiques les plus informatives ou discriminantes dans un ensemble de données ;

— Amélioration de la qualité d'image.

Nous nous intéressons, dans la suite de cette section, à l'application des techniques DL pour les problèmes de régression.

3.3.2 Deep Learning pour la régression

La régression dans le contexte du DL se réfère à une tâche d'apprentissage supervisé où l'objectif est de prédire des variables cibles continues à partir d'un ensemble de variables d'entrée. Dans cette application, les modèles de DL, sont entraînés sur un ensemble de données contenant des exemples avec des variables d'entrée et des valeurs cibles associées. Le modèle apprend à partir de ces exemples en ajustant ses paramètres internes pour minimiser l'erreur de prédiction entre ses sorties estimées et les valeurs cibles réelles. Une fois entraîné, le modèle est capable de prédire avec précision la valeur cible pour de nouvelles données d'entrée. L'application du DL pour la régression implique généralement les étapes suivantes :

1. Collecte des données : collecter un ensemble de données contenant des exemples avec des variables d'entrée et des variables cibles continues à prédire ;
2. Prétraitement des données : cela peut inclure des étapes telles que le nettoyage des données pour supprimer les valeurs aberrantes ou manquantes, la normalisation des caractéristiques pour mettre les données à l'échelle, et la division des données en ensembles d'entraînement, de validation et de test ;
3. Construction du modèle : la création d'un réseau de neurones avec une ou plusieurs couches cachées. La taille et la complexité du modèle peuvent varier en fonction de la nature du problème et de la quantité de données disponibles ;

4. Entraînement du modèle : le modèle est ensuite entraîné sur le dataset d'entraînement en utilisant un algorithme d'optimisation tel que la descente de gradient stochastique (Stochastic Gradient Descent (SGD)) ou ses variantes. Pendant l'entraînement, le modèle ajuste ses poids et ses biais pour minimiser une fonction de perte qui mesure l'écart entre les prédictions du modèle et les valeurs réelles de la variable cible ;
5. Validation du modèle : le modèle entraîné est évalué sur un ensemble de données de validation distinct pour évaluer ses performances et détecter tout surajustement (overfitting) ;
6. Test et évaluation du modèle : le modèle est évalué sur un ensemble de données de test indépendant pour estimer ses performances en conditions réelles. Cela permet de vérifier si le modèle généralise bien sur de nouvelles données et de comparer ses performances avec d'autres modèles ou approches.

En suivant ces étapes, les applications de DL pour la régression peuvent produire des modèles précis capables de prédire avec précision des variables continues à partir de données d'entrée. Les capacités de ces techniques se basent sur plusieurs fondements théoriques, présentés dans la sous-section suivante.

3.3.3 Fondements théoriques du Deep Learning

Cette section est consacrée pour représenter de manière simplifiée les fondements théoriques des techniques DL.

3.3.3.1 Les neurones artificiels et les réseaux de neurones

Les neurones artificiels et les réseaux de neurones sont les éléments de base du DL :

— **Neurone artificiel** : c'est une unité de traitement fondamentale qui prend plusieurs entrées pondérées, les combine linéairement, applique une fonction d'activation à la somme pondérée, et produit une sortie. La sortie peut être utilisée comme entrée pour d'autres neurones ou comme résultat final du modèle. La figure 3.7 représente une vue simplifiée d'un neurone artificiel. Soit $x = (x_1, x_2, ..., x_n)$ le vecteur d'entrée, $w = (w_1, w_2, ..., w_n)$ le vecteur des poids associés à chaque entrée, b le biais et f la fonction d'activation. Le neurone artificiel calcule la sortie y comme suit :

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

— **Réseau de neurones** : c'est une structure composée de plusieurs couches de neurones connectées. Les neurones sont organisés en couches d'entrée, de sortie et intermédiaires appelées

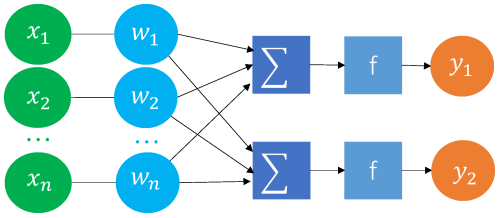


FIGURE 3.7 – Une vue simplifiée d'un neurone artificiel

couches cachées. Chaque neurone dans une couche est connecté à tous les neurones de la couche précédente et de la couche suivante. Le calcul des sorties dans un réseau de neurones se fait de manière itérative en propageant les entrées à travers le réseau jusqu'à la couche de sortie. Soit $x^{(l)}$ le vecteur d'entrée de la couche l, $w^{(l)}$ la matrice des poids associée à la couche l, $b^{(l)}$ le vecteur de biais associé à la couche l et $f^{(l)}$ la fonction d'activation de la couche l. La sortie $y^{(l)}$ de la couche l est calculée comme suit :

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

Les valeurs de sortie $y^{(l)}$ sont ensuite utilisées comme entrées pour la couche suivante jusqu'à ce que la sortie finale soit obtenue. La figure 3.8 représente une vue simplifiée d'un réseau de neurones à deux couches.

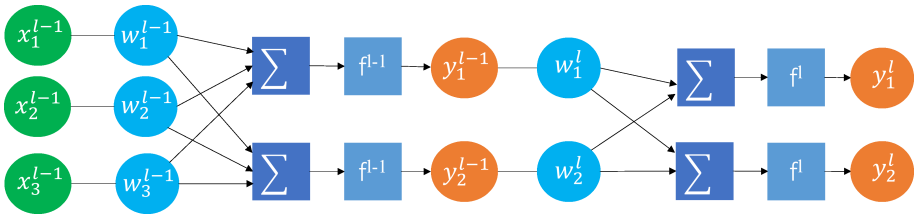


FIGURE 3.8 – Une vue simplifiée d'un réseau de neurones à deux couches

3.3.3.2 Le fonctionnement de l'entraînement et l'optimisation des paramètres

(a) **Principe général de l'entraînement** : Dans un modèle DL, il existe un ensemble de paramètres entraînables qui sont ajustés et optimisés au fil de plusieurs itérations afin de réduire au minimum une fonction de pertes (erreur) lors de l'étape d'entraînement. La figure 3.9 illustre les étapes de l'entraînement effectuées lors de ces itérations. Chaque itération est appelée une "epoch". En effet, pendant chaque "epoch", le processus d'entraînement itère sur l'ensemble des données, mettant à jour les poids du réseau pour minimiser la fonction de perte, généralement utilisée pour évaluer la différence entre les valeurs réelles et les valeurs prédites des sorties cibles. Pendant l'entraînement, le modèle est généralement évalué sur un ensemble de données distinct appelé ensemble de

validation, qui n'est pas utilisé pour l'entraînement lui-même. La validation permet de surveiller la capacité du modèle à généraliser correctement à de nouvelles données, en fournissant une estimation de la performance du modèle sur des données qu'il n'a pas encore vues. La validation se fait souvent à intervalles réguliers, par exemple à la fin de chaque "epoch". Pour cela, le nombre d'epochs est un hyperparamètre crucial qui influence la qualité de l'entraînement du modèle. Un autre hyperparamètre nécessitant un choix judicieux est la taille d'un "batch". En effet, les données sont généralement divisées en "batches" de taille fixe pour être traitées de manière plus efficace. Un "batch" est un sous-ensemble des données d'entraînement utilisé pour optimiser et mettre à jour les poids du réseau à chaque itération. Les poids du modèle à entraîner sont généralement initialisés avec des valeurs aléatoires. Ensuite, à la fin du traitement de chaque batch, ces poids sont mis à jour pour minimiser la fonction de pertes, en appliquant un algorithme d'optimisation. Nous nous intéressons à la suite aux optimiseurs les plus couramment utilisés.

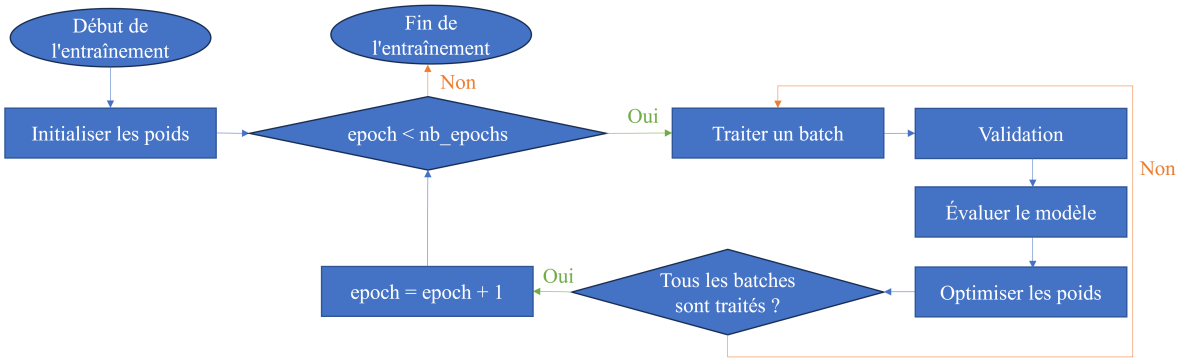


FIGURE 3.9 – Organigramme d'entraînement d'un modèle DL

(**b**) **Les techniques d'optimisation et d'ajustement des paramètres** : L'optimisation des poids d'un réseau de neurones pour un modèle DL est un aspect crucial pour obtenir des performances optimales. À la suite, les optimiseurs les plus couramment utilisés sont décrits. Dans cette description : w sont l'ensemble de poids à optimiser, L est la fonction de pertes, α est le taux d'apprentissage (learning rate) et $\nabla L(w_t)$ est le gradient de la fonction de pertes par rapport aux paramètres w évalué à l'itération t.

— Le Gradient Descent (GD) : Son principe de base consiste à mettre à jour les paramètres (les poids) dans la direction opposée au gradient de la fonction de pertes par rapport aux paramètres. Mathématiquement, la mise à jour des paramètres w à l'itération $t+1$ est donnée par :

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

Le taux d'apprentissage (α) contrôle la taille des pas de mise à jour des paramètres. Un taux d'apprentissage trop petit peut entraîner une convergence lente, tandis qu'un taux d'apprentissage trop grand peut entraîner une instabilité et un risque de divergence.

— Le SGD : est une variante de GD qui utilise un petit lot de données à la fois pour mettre à jour les poids du modèle. À chaque itération t, un exemple d'entraînement (x_i, y_i) est échantillonné aléatoirement à partir du dataset d'entraînement. Les paramètres du modèle sont alors mis à jour selon l'équation 3.21 :

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

SGD est largement adopté comme l'un des algorithmes d'optimisation les plus populaires pour entraîner des modèles DL, en raison de sa simplicité de mise en oeuvre et de son efficacité élevée [91].

— Le Batch Gradient Descent (BGD) [92] : est une variante de GD qui utilise l'ensemble complet des données du batch pour calculer le gradient de la fonction de pertes et mettre à jour les paramètres. Mathématiquement, la mise à jour des poids w à l'itération $t+1$ avec le BGD est donnée par l'équation 3.22 :

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

où $|B|$ est la taille du batch.

BGD peut être plus précis que SGD, mais il peut être plus lent en raison de la nécessité de calculer le gradient sur tout l'ensemble des données du batch.

— Le Follow the Regularized Leader (FTR) : utilise une fonction de pertes régularisée $L_t(w)$ qui combine la fonction de pertes $\ell_t(w)$ avec une pénalité de régularisation $R(w)$ sur les poids du modèle : $L_t(w) = \ell_t(w) + R(w)$. La mise à jour des poids du modèle se fait avec l'équation 3.23 :

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

La combinaison de l'optimisation par GD avec la régularisation pour mettre à jour les poids du modèle rend FTR efficace pour gérer des ensembles de données de grande taille [93].

Ces algorithmes d'optimisation peuvent être utilisés avec diverses techniques de DL pour résoudre des problèmes de régression. La prochaine sous-section présente les principales de ces techniques.

3.3.4 Des techniques de DL pour la régression

Pour les problèmes de régression en DL, plusieurs techniques peuvent être utilisées, notamment :

— **Fully Connected Neural Networks** : un réseau se compose de plusieurs couches entièrement connectées, où chaque neurone dans une couche est connecté à tous les neurones de la couche précédente ainsi que de la couche suivante.

— **DNN** : un réseau comprend plusieurs couches, notamment une couche d'entrée, une couche de sortie et des couches intermédiaires (cachées). Les DNN peuvent gérer des relations non linéaires et sont capables de traiter de vastes quantités de données.

— **ANN** : sont des modèles informatiques inspirés du fonctionnement du cerveau humain, composés de neurones organisés en couches.

— **CNN** : sont principalement utilisés pour la vision par ordinateur, mais ils peuvent également être appliqués à des problèmes de régression lorsque les entrées ont une structure spatiale. Selon [94], les CNN sont la technique de DL la plus populaire en raison de leur efficacité dans le traitement de données volumineuses.

— **Recurrent Neural Network (RNN)** [95] : les connexions entre les noeuds peuvent être récursives, ce qui signifie que l'entrée de certains noeuds dépend de leur propre sortie précédente. Cette caractéristique permet aux RNN de traiter efficacement des données séquentielles.

— **Long Short-Term Memory (LSTM)** : une version des RNN, spécialement conçue pour traiter efficacement les dépendances à long terme dans les séquences.

— **Gated Recurrent Unit network (GRU)** : une variante des RNN, caractérisée par un nombre réduit de paramètres et une convergence rapide. Son développement a été présenté dans [96].

— **Auto-encoder Neural Network (AENN)** : sont utilisés pour apprendre des représentations compressées des données d'entrée. Ils peuvent être utilisés pour la réduction de dimensionnalité et la reconstruction de données, ce qui peut être utile dans certains problèmes de régression.

— **MLP** : les connexions se font uniquement de l'entrée vers la sortie (en une seule direction). Il n'inclut aucune récurrence, ce qui signifie que la sortie d'un neurone ne peut pas influencer son prochain input.

— **Ensemble learning** : Cette approche consiste à fusionner les sorties de deux ou plusieurs modèles afin d'améliorer les résultats obtenus. Par exemple, certaines méthodes calculent la moyenne des sorties fournies par plusieurs modèles de DL et de ML.

Le choix de la technique de DL à employer doit être réalisé selon les exigences spécifiques du problème ainsi que la nature des données, afin d'obtenir des performances optimales lors de la résolution des problèmes de régression. Plusieurs critères sont disponibles pour évaluer ces performances, et la sous-section suivante répertorie les critères les plus couramment utilisés.

3.3.5 Les critères de performance des techniques récentes de DL

Les techniques de DL pour la régression peuvent être évaluées selon différents critères de performance. Ces derniers peuvent varier en fonction des besoins spécifiques du problème et des données disponibles. Les critères d'évaluation couramment utilisés sont décrits à la suite. Dans les équations des critères, y_i désigne la valeur réelle, $\hat{y}_i$ représente la valeur prédite par le modèle de DL, et n est le nombre total de points de mesure.

— MAE : il s'agit de la moyenne des valeurs absolues des écarts entre les valeurs prédites et les valeurs réelles. MAE mesure la taille moyenne des erreurs de prédiction sans tenir compte de leur direction.

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

— Mean Squared Error (MSE) : il s'agit de la moyenne des carrés des écarts entre les valeurs prédites et les valeurs réelles. MSE pénalise les grandes erreurs.

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

— Root Mean Squared Error (RMSE) : il s'agit de la racine carrée de la moyenne des carrés des écarts entre les valeurs prédites et les valeurs réelles.

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

— R-squared (R2) : cela quantifie la proportion de la variance des valeurs prédites par les valeurs réelles (Équation 3.27). Il est compris entre 0 et 1, où 1 indique un ajustement parfait du modèle aux données.

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

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

— Mean Absolute Relative Error (MARE) : il s'agit de la moyenne des valeurs absolues des écarts relatifs entre les valeurs prédites et les valeurs réelles, exprimées en pourcentage.

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

— Accuracy : représente le nombre de prédictions correctes faites par le modèle par rapport au nombre total de prédictions.

$$Accuracy = \frac{Nombre\ de\ prdictions\ correctes}{Nombre\ total\ de\ prdictions} \times 100\% \tag{3.29}$$

— Recall : est une mesure de performance utilisée en classification pour évaluer la capacité du système à capturer et identifier correctement les objets pertinents. Cette mesure peut être adaptée pour la régression selon l'équation 3.30.

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

nb_c_pred est le nombre de prédictions correctes (le nombre de valeurs pertinentes correctement prédites) et $tot\_nb\_succ\_pred$ représente le nombre total de valeurs pertinentes à prédire.

— F1-score : il s'agit d'une moyenne harmonique de l'accuracy et du recall, et fournit une mesure unique qui équilibre le compromis entre ces deux métriques 3.31.

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

3.3.6 Comparaison et revue des travaux récents optimisant des techniques DL par métaheuristiques

Dans le domaine de la recherche, une tendance émerge visant à optimiser les techniques de DL à l'aide de métaheuristiques. Le tableau 3.2 décrit des travaux récents abordant cette approche.

TABLEAU 3.2: Comparaison et revue des travaux récents optimisant des techniques DL par métaheuristiques

Papier	Année	Technique DL	Métaheuristique	Objectif de l'optimisation	Problème étudié	Résultats sans l'optimisation	Résultats avec l'optimisation
[97]	2023	CNN	"Particle Swarm Optimization without Velocity" (PSWV)	optimiser les hyperparamètres	Classification d'images	-	Taux d'erreurs : 3.64%
[98]	2023	LSTM	Artificial Rabbits Optimization Algorithm (ARO) [99]	optimiser les hyperparamètres	Prédiction du prix des actions	MSE : 44.44 , MAE : 5.21 , R2 : 82.9%	MSE : 24.287, MAE : 3.804 , R2 : 90.7%
[100]	2022	DNN	IPSO : une version améliorée de PSO	optimiser les hyperparamètres	L'impact de la distanciation sociale sur la propagation du COVID-19	MAE : 6.1002 RMSE : 49.3345	MAE : 4.8177, RMSE : 45.0471
[101]	2022	CNN+LSTM	Whale Optimization Algorithm (WOA)[102]	optimiser des hyperparamètres	Reconnaissance des émotions dans la parole	Accuracy : 92.24%	Accuracy : 99.47%
[103]	2022	CNN	GWO	optimiser les hyperparamètres	Détection du cancer de la peau	-	Accuracy : 81.46%
[104]	2021	CNN	Binary PSO : une variante de PSO	identifier les caractéristiques les plus pertinentes et réduire la dimensionnalité des données	un "Intrusion Detection System" (IDS) intelligent	-	Accuracy : 94.3%, Reacall : 91%
[105]	2020	DNN	GWO	réduire les données : réduire la dimension des attributs en sélectionnant les attributs hautement influents	Détection d'intrusion dans une architecture IoMT (Internet of Medical Things)	Accuracy : 99.2%	Accuracy : 99.9%
[106]	2020	DNN	un algorithme génétique	optimiser les hyperparamètres	Prédiction de la consommation énergétique dans les bâtiments	RMSE : 11.83	RMSE : 10.29
[107]	2020	ANN	PSO	optimiser les hyperparamètres et les valeurs initiales des poids	Prédiction du point de puissance maximale d'un ensemble photovoltaïque	Efficacité : 98.3%	Efficacité : 99.49%
[108]	2019	CNN	cPSO : une nouvelle variante de PSO proposée	optimiser les hyperparamètres	analyse d'ablation et comparaison de travaux similaires	-	Taux d'erreur : 8.67 %

Synthèse L'étude des travaux récents visant à optimiser les techniques de DL à l'aide de métaheuristiques 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.

3.4 Comparaison et revue des études récentes résolvant le problème du 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, le tableau 3.3 présente une revue des travaux récents. Afin d'éviter de répéter les descriptions de certains travaux déjà exposés dans le tableau 2.2 (la section 2.3.3), ils sont simplement référencés dans ce tableau. Ce dernier inclut des informations sur les algorithmes d'optimisation utilisés ainsi que les résultats obtenus, soulignant ainsi l'impact positif de ces techniques sur l'amélioration des performances.

TABLEAU 3.3: Comparaisons entre des travaux récents résolvant le problème de routage dans les réseaux IoT en se basant sur les algorithmes d'optimisation

Papier	Année	Type	Approche/Objectifs	Algorithme d'optimisation	Résultats	Inconvénients
[15]	2.2					
[17]	2.2					
[109]	2022	géographique	Sélectionner les routes en prenant en compte la position des noeuds, leur niveau d'énergie et les données sur le trafic	Bacteria Foraging Optimization (BFO) [110]	Délai : réduit de 12% ; Débit : augmenté de 10% ; énergie résiduelle : augmentée de 11% ; PDR : augmenté de 7 à 9%	Ne prendre pas en compte les noeuds mobiles ; 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 ; Simulations avec Matlab
[83]	2022	basé sur la QoS	Développer un algorithme de routage économe en énergie	AcNSGA-III : un nouvel algorithme d'optimisation proposé	Latence moyenne : 1.3s-2.8s ; Durée de vie moyenne : $\approx$ 19400 s	Pas de comparaisons à d'autres solutions de routage existantes ; Applicabilité à d'autres types de réseaux nécessiterait des investigations supplémentaires.
[111]	2022	réactif	minimiser la distance de transmission moyenne maximale entre les noeuds, tout en équilibrant la gestion de l'énergie	MOGWO	PDR : 98% ; Délai : réduit de 40% ; Consommation d'énergie : réduite de 30%	Pas de noeuds mobiles ; Pas d'expérimentations réelles ; Des retards supplémentaires dans la transmission des données ; Charge computationnelle plus élevée et temps de convergence plus long
[112]	2022	basé sur la QoS	baser le routage sur un nouvel algorithme d'optimisation et une nouvelle fonction fitness	Reposition PSO (RPSO) : nouvelle variante de PSO proposée	Durée de vie du réseau : 14% à 29% plus élevée ; Nombre de noeuds capteurs inactifs : réduit de 52% à 58% de moins ; Consommation d'énergie : réduite de 61% à 70% ; Débit : accru de 13% à 36%	Comparaison avec d'autres algorithmes d'optimisation plutôt qu'avec d'autres solutions de routage existantes ; Pas d'expérimentations réelles
[113]	2021	multi-chemins	Concevoir un nouvel algorithme d'optimisation hybride en intégrant le Sun Flower Optimization (SFO) [114] et GWO pour établir un routage multi-chemins	SFO et GWO	Délai : 0.779 s ; Durée de vie du réseau : 98.039 % ; Débit : 47.368%	Pas d'expérimentations réelles ; Une seule fonction objectif ;
[115]	2021	multi-chemins	calculer la fonction de fitness en utilisant la distance entre les noeuds dans le réseau	un algorithme génétique	Consommation d'énergie : 0.9-5.9J ; Taux de noeuds relais actifs : 30%	Pas de comparaisons avec d'autres solutions de routage ; S'intéresser seulement à la consommation d'énergie ; Pas d'expérimentations réelles

Suite à la page suivante

Tableau 3.3 – suite de la page précédente

Papier	Année	Type	Approche/Objectifs	Algorithme d'optimisation	Résultats	Inconvénients
[116]	2020	orienté données	résoudre le routage des données multimédias dans un réseau IoT	Shuffled Frog Leaping Optimization Algorithm (SFLA) [117]	PDR : ≈ 90% ; Débit moyen : ≈ 3.5 Mbps ; Durée de vie : 184 cycles	Pas d'expérimentations réelles ; La proposition ne prend pas en charge la redondance des données, un défi qui a un impact significatif sur les performances d'un réseau multimédia
[118]	2020	basé sur la QoS	développer un algorithme de routage pour les "Mobile Ad-hoc Networks"	Intelligent Water Drop Optimization Algorithm (IWDOA) [119] et Firefly Optimization Algorithm (FOA) [120]	PDR : 82.64% pour IWDOA et 82.26% pour FOA	Pas d'expérimentations réelles ; Ne prend pas en charge l'auto-stabilisation (agir automatiquement en cas de problèmes) et l'identification des noeuds malveillants, les deux défis les plus importants dans les réseaux IoT-MANET [121]
[122]	2020	basé sur la QoS	développer un protocole de routage pour les réseaux de capteurs sans fil hétérogènes	une variante de GWO proposée	Durée de vie du réseau : 1400-2300 cycles ; Améliorer le débit de 94.02%	Pas d'expérimentations réelles; Prendre en compte un seul objectif, la consommation d'énergie ; L'environnement étudié ne contient que la communication entre une station de base et les chefs d'étoiles; Pas de noeuds mobiles.
[123]	2020	orienté-données	développer Content Awareness Routing Algorithm for IoT Networks (CARA-IoT), un algorithme de routage dans les réseaux de l'IoT Multimédias	ACO	PDR : 70%-80%	Pas d'expérimentations réelles ; Ne s'intéresse pas au problème de la redondance des données qui représente un défi significatif dans les réseaux Multimédia
[124]	2020	basé sur la QoS	sélectionner les routes en fonction de l'énergie résiduelle et du niveau de confiance pour un routage sécurisé et fiable	PSO et l'algorithme "Salp Swarm Optimization" (SSO) [125]	Délai : 0.08ms ; Taux de détection : 83%	Pas d'expérimentations réelles
[126]	2019	basé sur la QoS	Réduire la probabilité de piratage	Imperialist Competitive Optimization Algorithm (ICOA)[127]	Confidentialité : 0.72 ; Intégrité : 0.70 ; Disponibilité : 0.43	Pas d'expérimentations réelles ; Les tests sont menés dans un réseau maillé de petite taille (20 noeuds) ; Les résultats présentés n'incluent pas de comparaisons avec d'autres protocoles.

Synthèse L'optimisation est un paradigme largement appliqué avec succès dans de nombreux domaines, et il est donc impératif de tirer parti de ce succès pour étudier le routage dans les réseaux IoT. Cette approche pourrait permettre d'améliorer les performances et de remédier à la complexité inhérente à cette tâche. L'examen des travaux antérieurs 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 mentionnés précédemment 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.

3.5 Comparaison et revue des études récentes résolvant le 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. Le tableau 3.4 présente des détails des travaux récents menés dans ce domaine. Pour éviter de dupliquer les descriptions des travaux déjà présentées dans le tableau 2.3 (la section 2.4.2), ils sont simplement référencés dans ce tableau. Ce dernier met en évidence les métaheuristiques et/ou les techniques DL utilisées, ainsi que les résultats obtenus pour illustrer leur contribution à l'amélioration des performances pour relever ce défi.

TABLEAU 3.4: Comparaisons entre des travaux récents résolvant le problème de localisation en intérieur dans les réseaux IoT en se basant sur les métaheuristiques et/ou les techniques DL

Papier	Année	Métaheuristique	DL	Type	Approche/Objectifs	Techniques	Technologie	Mesures	Résultats		Inconvénients
									MAE	Précision	
[43]	2023	✗	✓	2.3							
[31]	2023	✓	✗	2.3							
[32]	2023	✓	✗	2.3							
[44]	2023	✗	✓	2.3							
[128]	2023	✓	✗	range-based	Développer une solution de localisation et du suivi d'un cible	PSO	-	RSSI	-	93.09%	Pas de comparaisons avec d'autres solutions existantes; Pas d'expérimentations réelles
[46]	2022	✗	✓	2.3							
[129]	2022	✓	✗	range-based	résoudre le problème de localisation des noeuds "edge computing"	GWO et Moth-Flame Optimization (MFO) [130]	-	RSSI	<1m pour GWO et <2m pour MFO	-	Comparaison avec d'autres métaheuristiques plutôt qu'avec des solutions de localisation existantes; Les mesures RSSI peuvent être affectées par des facteurs environnementaux (bruit ou interférences); Pas d'expérimentations réelles
[131]	2022	✓	✗	range-free	Proposer un algorithme d'optimisation hybride et l'intégrer à l'algorithme DV-HoP	"Tunicate Swarm Algorithm" [132] et "Harris hawk optimization" [133]	-	-	0.45 m	-	L'hybridation de deux algorithmes métaheuristiques peut augmenter la complexité de l'algorithme de localisation; Le réglage des paramètres peut s'avérer difficile et nécessite du temps; Pas d'expérimentations réelles (Matlab)
[134]	2022	✓	✗	range-based	développer une approche hybride combinant les points forts de la version améliorée de PSO, le "Inertial Measurement Unit" (IMU) et le RSSI fingerprinting	une version améliorée de PSO proposée	Wi-Fi	RSSI	0.7m	-	La complexité de la combinaison de 3 méthodes; Les mesures RSSI peuvent être affectées par des facteurs environnementaux; La solution nécessite des informations cartographiques précises (peut être difficile et prendre du temps)
[48]	2021	✗	✓	2.3							

Suite à la page suivante

Tableau 3.4 – suite de la page précédente

Papier	Année	Métaheuristique	DL	Type	Approche/Objectifs	Techniques	Technologie	Mesures	Résultats		Inconvénients
									MAE	Précision	
[49]	2021	✗	✓	2.3							
[135]	2021	✓	✗	range-based	Mettre à jour de manière itérative les positions des noeuds inconnus en fonction de leurs distances aux ancres, jusqu'à ce que les positions convergent vers un minimum local	Group Teaching Optimization Algorithm (GTOA) [136]	-	RSSI	-	64.5-94%	Comparaison avec d'autres métaheuristiques plutôt qu'avec des solutions de localisation existantes ; Les mesures RSSI peuvent être affectées par des facteurs environnementaux ; Pas d'expérimentations réelles (Matlab)
[55]	2020	✗	✓	2.3							
[137]	2020	✓	✗	range-based	Développer une solution de localisation pour les systèmes de stationnement intelligent : minimiser l'erreur de localisation et réduire le nombre requis d'ancres	MOGWO	-	RSSI	RMSE ≈ 0.42-0.52m	67%-93%	Ne prendre pas en compte un environnement dynamique (des véhicules mobiles) ;Pas d'expérimentations réelles ; Les mesures de RSSI peuvent être affectées par divers facteurs

Synthèse De récentes recherches explorent l'utilisation de métaheuristiques et/ou de techniques DL pour développer des solutions de localisation en intérieur dans les réseaux IoT. L'étude présentée dans cette section montre que 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 encore 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. En effet, cette étude confirme la synthèse 2.4.3. 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.

3.6 Conclusion

En conclusion, ce chapitre a exploré deux domaines cruciaux offrant des perspectives précieuses sur la résolution de problèmes complexes. Nous avons examiné les différents types de problèmes d'optimisation, les méthodes de résolution associées et les critères de performance utilisés pour évaluer ces méthodes. Nous avons également exploré le domaine du DL, en examinant ses fondements théoriques, ses applications pratiques et les critères de performance associés. Enfin, nous avons mis en évidence la convergence entre l'optimisation et le DL, en étudiant des travaux qui utilisent les métaheuristiques pour optimiser les techniques de DL, ouvrant ainsi la voie à des solutions plus efficaces pour résoudre une variété de problèmes réels. Des revues concernant l'application de ces techniques avantageuses pour résoudre les problèmes de routage et de localisation dans les réseaux IoT ont été étudiées. En résumé, ce chapitre offre un aperçu complet des concepts, des méthodes et des tendances de recherche dans les domaines de l'optimisation et du DL, offrant ainsi une base pour la suite de nos contributions dans le contexte de résolution de problèmes complexes de routage et de localisation dans les réseaux IoT.

Deuxième partie

Contributions

Chapitre 4

Contributions théoriques

Sommaire

4.1 Introduction

Ce chapitre propose une présentation approfondie de nos contributions théoriques, qui incluent une modélisation mathématique du routage dans les réseaux IoT ainsi que la proposition de l'algorithme IMOGWO, une amélioration de l'algorithme MOGWO existant. Nous détaillons également l'optimisation des paramètres des modèles DL à l'aide de métaheuristiques. De plus, une section est consacrée à notre proposition de localisation en intérieur dans les réseaux IoT, combinant les techniques DL et les métaheuristiques. En tirant parti des approches précédentes, nous concluons ce chapitre en détaillant notre modèle de routage géographique hybride et multi-objectifs, destiné à être appliqué dans les réseaux IoT à architecture maillée.

4.2 Modélisation mathématique du routage dans un réseau IoT

Nous proposons, dans cette section, une modélisation mathématique du routage dans un réseau IoT. Cette modélisation permet de saisir la structure et les caractéristiques du réseau IoT à prendre en compte dans nos contributions. Elle contribue également à formaliser les concepts et les relations entre les objets connectés dans ce réseau, ce qui rend les approches de routage proposées dans les sections suivantes plus structurées et facilite la compréhension de leur fonctionnement.

4.2.1 La notation

La modélisation mathématique proposée considère les ensembles, les variables de décision et les paramètres suivants :

— N : Le réseau considéré.

— O : L'ensemble des objets connectés dans N.

— L : L'ensemble des liaisons entre les objets connectés.

— $V(o)$: L'ensemble des voisins de l'objet o.

— $V2S(o)$: L'ensemble des voisins à 2 sauts de l'objet o.

— o_s : La source initiale d'une transmission.

— o_d : La destination finale d'une transmission.

— o_c : L'objet courant lors du routage (où le message arrive pendant l'acheminement).

— o_n : Un objet voisin du o_c.

— I_o : Un identifiant unique à chaque objet o dans le réseau.

— (x_o, y_o) : Les coordonnées d'un objet o.

— (x_s, y_s) : Les coordonnées de la source initiale o_s.

— (x_d, y_d) : Les coordonnées de la destinataion finale o_d.

— $F(o)$: Un degré de fiabilité à chaque objet o dans le réseau.

— $T_v(o)$: La table de voisinage d'un objet o.

— $Distance(o_1, o_2)$: La distance entre deux objets voisins o_1 et o_2.

— d_{ref} : Une distance de référence entre deux objets voisins.

— E_{init} : L'énergie initiale de chaque objet.

— E_{Tx} : La consommation d'énergie d'émission.

— E_{Rx} : La consommation d'énergie de réception.

— E_{Sleep} : La consommation d'énergie de sommeil.

— E_{Listen} : La consommation d'énergie d'écoute.

— E_{residual} : L'énergie résiduelle de chaque objet.

— $E^i_{\text{Tx\_bit}}$: L'énergie consommée par un objet i lorsqu'il émet un bit.

— $E^i_{\text{Rx\_bit}}$: L'énergie consommée par un objet i lorsqu'il reçoit un bit.

— T_m : La taille d'un message à émettre en bits.

— T_{mc} : La taille d'un message de contrôle à émettre en bits.

— $Nb\_sauts(route)$: Le nombre de sauts dans une route de routage.

— R : Une route de routage, qui est une liste d'identifiants d'objets dans le réseau.

— $DLModel$: est le modèle DL de localisation entraîné.

— S_i : une étoile i.

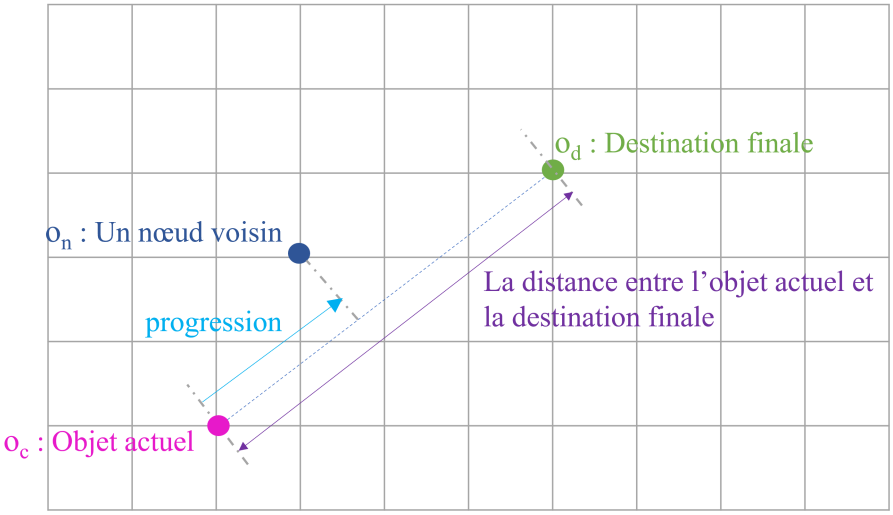
— H_i : le chef de l'étoile i.

— B : une passerelle.

4.2.2 Les objectifs

L'objectif principal des fonctions objectifs dans le contexte du problème de routage IoT est d'optimiser le coût global de chaque itinéraire et de sélectionner le meilleur chemin parmi les options disponibles. Ce coût est calculé en tenant compte de la distance entre les objets parcourus lors du routage, des performances énergétiques, de l'état du trafic qui va influencer la latence et d'autres métriques pertinentes. Les objectifs à considérer pendant ce routage multi-saut sont décrits dans les sous-sections suivantes.

FIGURE 4.1 – La progression positive du noeud actuel dans la route vers la destination finale



4.2.2.1 La progression positive vers la destination finale

— Objectif : cette fonction vise à avoir la plus grande progression positive vers la destination finale o_d, partant de l'objet courant o_c et en passant par un objet voisin o_n, et se rapprocher le plus de celle-ci (figure 4.1). Le critère utilisé pour évaluer cette progression est le pourcentage d'avancement vers la destination.

— Fonction objectif :

$$Maximiser \frac{Progression \times 100}{Distance(o_c, o_d)}$$
$$\text{Où } Progression = \frac{D(o_c, o_d)^2 + D(o_c, o_n)^2 - D(o_n, o_d)^2}{2 \times D(o_c, o_d)} \tag{4.1}$$

D est une fonction qui retourne la distance entre deux objets pris en paramètres.

Démonstration : En appliquant le théorème de Pythagore pour les longueurs des côtés dans un triangle rectangle : $D(o_c, o_n)^2 - Progression^2 = D(o_n, o_d)^2 - (D(o_c, o_d) - Progression)^2$
Ce qui donne : $2 \times D(o_c, o_d) \times Progression = D(o_c, o_d)^2 + D(o_c, o_n)^2 - D(o_n, o_d)^2$

Exemple Soient $(2, 1)$, $(3, 3)$ et $(6, 4)$ les coordonnées de o_c, o_n et o_d, respectivement. $D(o_c, o_d) = 5$
$D(o_c, o_n) = 2.24$
$D(o_n, o_d) = 3.16$
Donc $Progression = \frac{5^2 + 2.24^2 - 3.16^2}{2 \times 5} = 2.0032$

4.2.2.2 Gestion efficiente de l'énergie

Afin d'avoir une gestion efficiente de l'énergie, les objectifs suivants sont considérés :

La consommation énergétique

— Objectif : cette fonction vise principalement à optimiser la consommation d'énergie moyenne des objets IoT lors de la réception et de la transmission des messages sur un seul saut. À chaque étape du routage, un objet intermédiaire o_c, situé entre l'émetteur initial et la destination finale, reçoit une quantité T_m de données avant de les transmettre à un autre objet, o_n, situé à une distance d. Ce processus doit respecter deux contraintes : d'une part, la consommation d'énergie de o_c ne doit pas dépasser son niveau d'énergie résiduelle, et d'autre part, o_n doit disposer d'une énergie résiduelle suffisante pour recevoir et renvoyer les données à o_c en cas de besoin.

— Fonction objectif :

$$\begin{aligned} &Minimiser \sum E^i_{\text{Rx}} + E^i_{\text{Tx}}(d) \\ &\text{Où} \quad E^i_{\text{Rx}} = E^i_{\text{Rx}} + T_m \times E^i_{\text{Rx\_bit}} \quad \text{et} \quad E^i_{\text{Tx}}(d) = E^i_{\text{Tx}} + T_m \times E^i_{\text{Tx\_bit}} + m \times d^2 \\ &\text{Sous contraintes : } E^c_{\text{Rx}} + E^c_{\text{Tx}}(d) \leq E_{\text{residual}}(o_c) \quad \text{et} \quad E^n_{\text{Rx}} + E^n_{\text{Tx}}(d) \leq E_{\text{residual}}(o_n) \end{aligned} \tag{4.2}$$

L'écart-type énergétique

— Objectif : cette fonction vise à minimiser l'écart-type énergétique (Standard-Deviation of Energy (SDE)), qui représente la variation de l'énergie totale consommée dans le réseau IoT. Il se réfère à la manière dont les niveaux d'énergie consommée par les différents objets du réseau diffèrent les uns des autres, ce qui permet d'évaluer l'équilibre de la consommation d'énergie dans le réseau, où une faible variation indique une distribution équilibrée de l'énergie, tandis qu'une variation plus élevée peut signaler des différences dans la consommation d'énergie entre les objets du réseau.

— Fonction objectif :

$$\begin{aligned} &Minimiser\ SDE(N) \\ &\text{Où} \quad SDE(N) = AVG(VAR(\sum E_{\text{Tx}} + E_{\text{Rx}} + E_{\text{Sleep}} + E_{\text{Listen}})) \end{aligned} \tag{4.3}$$

L'efficacité énergétique

— Objectif : l'efficacité énergétique, dans notre modélisation, consiste à gérer efficacement l'énergie restante dans les objets après la consommation. Il s'agit de minimiser le pourcentage d'énergie nécessaire à un objet pour recevoir et retransmettre le message par rapport à

son énergie résiduelle. Le critère utilisé est le pourcentage d'énergie à consommer par rapport à l'énergie résiduelle.

— Fonction objectif :

$$\begin{aligned} &Minimiser\ \frac{(E^c_{\text{Rx}} + E^c_{\text{Tx}}(d_{ref})) \times 100}{E_{\text{residual}}(o_n)} \\ &\text{Où} \quad E^c_{\text{Rx}} = E^c_{\text{Rx}} + T_m \times E^c_{\text{Rx\_bit}} \text{ et } E^c_{\text{Tx}}(d_{ref}) = E^c_{\text{Tx}} + T_m \times E^c_{\text{Tx\_bit}} + m \times d^2_{ref} \end{aligned} \tag{4.4}$$

La durée de vie du réseau

— Objectif : la durée de vie du réseau correspond à la période pendant laquelle au moins un objet reste opérationnel dans le réseau. Cette fonction sert à maximiser la durée de vie du réseau. Pour prolonger au maximum cette durée, il est nécessaire de répartir équitablement la charge des communications de routage entre les objets afin de retarder autant que possible l'arrêt du premier objet. Cette prolongation peut être anticipée en calculant la différence entre l'énergie initiale de tous les objets et la somme de l'écart-type énergétique et de la moyenne de l'énergie consommée par les objets IoT.

— Fonction objectif :

$$\begin{aligned} &Maximiser \quad (\sum_{o_i \in O} E_{\text{init}}(o_i) - (SDE(N) + C_{avg}(O)) \\ &\text{Où } C_{avg}(O) = \frac{\sum_{o_i \in O} E_{\text{Tx}}(o_i) + E_{\text{Rx}}(o_i) + E_{\text{Sleep}}(o_i) + E_{\text{Listen}}(o_i))}{|O|} \end{aligned} \tag{4.5}$$

4.2.2.3 La fiabilité globale

— Objectif : la fiabilité globale consiste à maintenir des liaisons fiables entre les objets, assurant ainsi des communications efficaces et continues. Dans cette modélisation, la fiabilité se base sur deux degrés : un degré de fiabilité et un degré de confiance pour chaque objet. Le degré de fiabilité est défini comme la probabilité que le capteur fonctionne correctement. Le degré de confiance représente le niveau de confiance accordé à un objet suite à des expériences précédentes.

— Fonction objectif :

$$\begin{aligned} &Maximiser \sum_{i=1}^{|O|} (Fiabilité_i \times Confiance_i) \\ &\text{Où } Fiabilité_i = \frac{temps\_de\_fonctionnement(o_i)}{temps\_total\_d\_observation} \end{aligned} \tag{4.6}$$

$Temps\_de\_fonctionnement(o_i)$ est le temps pendant lequel le capteur o_i est opérationnel, et $temps\_total\_d\_observation$ est la durée totale d'observation.

4.2.2.4 La continuité

La connectivité locale

— Objectif : la connectivité locale est interprétée dans cette modélisation comme le nombre de voisins qu'un noeud dispose à une distance d'un saut. Une meilleure connectivité peut garantir la continuité de la route vers la destination finale en réduisant le risque des vides. Le problème de vide se produit lorsqu'un paquet arrive à un objet qu'il n'a aucun autre voisin.

— Fonction objectif :

$$Maximiser|V(o_n)| \tag{4.7}$$

Voisins avantageux

— Objectif : cela vise à avoir des candidats avantageux parmi le voisinage pour le prochain saut en termes de connectivité, progression positive vers la destination finale et degré de fiabilité.

— Fonction objectif :

$$\begin{aligned} &Maximiser|V2S(o_n)| \\ &Maximiser \max_{o_i \in V2S} Progression(o_i, o_d) \\ &Maximiser \max_{o_i \in V2S} F(o_i) \end{aligned} \tag{4.8}$$

4.2.2.5 La latence de transmission

— Objectif : Un objectif de minimiser la latence de transmission consiste à réduire le temps écoulé entre l'émission d'une donnée depuis la source initiale et sa réception par la destination finale. Dans cette modélisation, nous considérons que la latence de transmission peut être affectée principalement par la distance physique.

— Fonction objectif :

$$\begin{aligned} &Minimiser \sum T(o_s, o_d) \\ &\text{Où } T(o_s, o_d) \text{ est le temps de transmission entre } o_s \text{ et } o_d \end{aligned} \tag{4.9}$$

Cette modélisation est utilisée pour modéliser nos contributions concernant le routage dans un réseau IoT. En premier lieu, nous utilisons MOGWO pour développer un routage réactif et multi-

objectifs. En deuxième lieu, nous proposons un routage géographique, hybride et multi-objectifs basé sur une version améliorée de MOGWO que nous proposons. La section suivante représente la première contribution concernant le routage dans un réseau IoT.

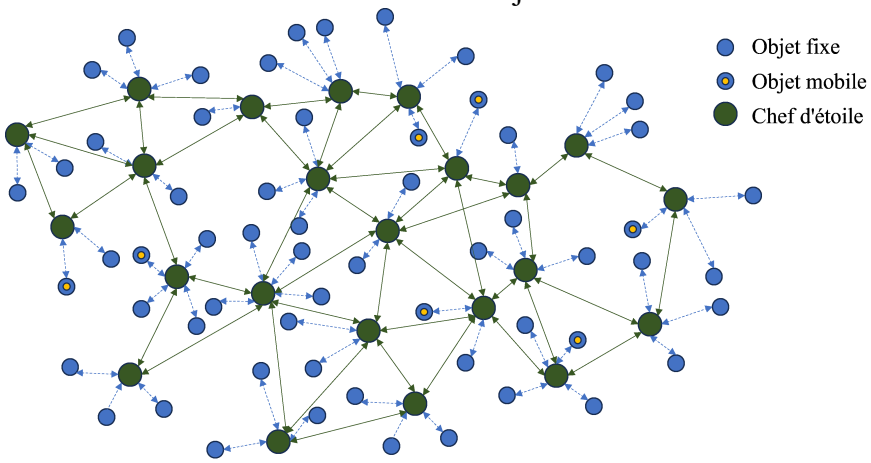
4.3 Un routage basé sur MOGWO

Pour cette contribution, nous proposons de bénéficier de l'algorithme MOGWO pour modéliser un routage réactif et multi-objectifs dans un environnement IoT. Cette idée provient du fait que GWO est une des méta-heuristiques récentes qui a gagné un très grand intérêt dans plusieurs domaines. Sa capacité de traiter des problèmes multi-objectifs dans sa version MOGWO, permet de modéliser et développer le routage visé. Cette section expose le modèle de réseau ainsi que les hypothèses à prendre en compte, en plus de l'algorithme de routage à élaborer.

4.3.1 Le modèle de réseau

Le modèle de réseau IoT que nous envisageons est illustré par la figure 4.2. Il 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. Les objectifs pris en compte sont ceux présentés dans la section 4.2.2, qui sont liés à la gestion efficace de l'énergie et l'optimisation 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 la destination finale. Ensuite, l'ordinateur

FIGURE 4.2 – L'architecture des objets connectés considérée



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.
Ce modèle de réseau offre ainsi une infrastructure flexible et évolutive pour la communication au sein de l'environnement IoT. Il permet non seulement les échanges entre les objets (Machine to Machine (M2M)), mais aussi avec les points de collecte et de traitement de données internes au réseau. De plus, il facilite les interactions avec des entités externes, favorisant ainsi l'intégration de l'environnement IoT avec d'autres systèmes et applications.

4.3.2 Les hypothèses

Les hypothèses suivantes sont prises en compte :

— Les chefs des étoiles disposent d'une quantité d'énergie initiale plus importante que les autres objets qui composent les étoiles. Ces chefs sont stationnaires (ne changent pas de positions).

— Les autres objets sont homogènes, ce qui signifie qu'ils ont la même quantité d'énergie initiale et la même portée de transmission.

— Le réseau contient des objets fixes et des objets mobiles.

— Wi-Fi est la technologie de communication utilisée dans le réseau. Ce réseau Wi-Fi est configuré en mode ad hoc, permettant ainsi une communication directe entre les différents objets sans nécessiter d'infrastructure centralisée.

— Chaque chef d'étoile doit maintenir des communications périodiques avec l'ordinateur central, ce qui lui permet d'avoir une vision globale actualisée du réseau, des niveaux d'énergie et de la disponibilité des routes de communication. Ce mécanisme garantit également une réactivité et une efficacité du routage lors de la transmission des données à travers le réseau.

4.3.3 L'algorithme de routage

Considérons un scénario où un objet communicant o_s, faisant partie de l'étoile S_s, souhaite transmettre un message m vers une destination D. Cette destination peut être un autre objet o_d appartenant à une étoile distincte S_d, ou une passerelle B menant à un réseau externe. H_s est le chef de l'étoile S_s, tandis que H_d est le chef de S_d. OC est l'ordinateur central qui est responsable du routage.
Pour résoudre l'acheminement du message, nous prenons en compte les éléments suivants :

— Le routage est considéré comme un problème d'optimisation.

— Les chefs des étoiles représentent les solutions dans l'espace de recherche.

— Le routage à effectuer consiste en des sauts multiples entre les chefs d'étoiles.

— À chaque étape de la résolution du problème d'optimisation, le prochain saut est déterminé, c'est-à-dire le prochain chef à atteindre.

— La solution retournée par l'algorithme d'optimisation MOGWO est la route à suivre pour l'acheminement.

— Les fonctions objectifs concernant la gestion efficace de l'énergie et la réduction de la latence de transmission, décrites dans la section 4.2.2, sont prises en compte.

Le processus de routage proposé est décrit par l'algorithme 8. Il se base sur MOGWO et suit plusieurs étapes pour acheminer un message à partir d'un objet o_s vers une destination finale H_d ou une passerelle B.
Initialement, l'objet o_s envoie le message m à H_s. Une fois que H_s reçoit le message, il vérifie si la destination finale, o_d, appartient à la même étoile (représentée par S_s) que lui. Si tel est le cas, H_s peut envoyer le message directement à o_d. Sinon, H_s doit demander à l'ordinateur central OC l'établissement d'une route optimisée, R, vers la destination finale.
Pour ce faire, OC commence par initialiser un ensemble d'agents de recherche, appelé Pop, avec les chefs des étoiles voisins. Ensuite, il calcule les valeurs des fonctions objectifs pour chaque agent de recherche et initialise une archive avec les solutions non dominées parmi ces agents. OC sélectionne ensuite les trois meilleures solutions parmi celles de l'archive. Une fois le prochain chef, X_α, identifié, il devient alors le prochain saut dans la route R.

Algorithme 8 : Algorithme de routage basé sur MOGWO

Data : Objet o_s, m, H_s, H_d, B
Result : La route R

Émettre(m, o_s, H_s)
if $o_d \in S1$ **then**
    Émettre(m, H_s, o_d)
else
    Émettre(demande_routage, H_s, OC)
    $X_{actuel} \leftarrow H_s$
    $R \leftarrow \{X_{actuel}\}$
    $Pop \leftarrow Voisins(H_s)$
    Évaluer(Pop)
    $archive \leftarrow solutionsNonDomines(Pop)$
    $X_\alpha, X_\beta, X_\delta \leftarrow$ meilleuresSolutions($archive$)
    $X_{actuel} \leftarrow X_\alpha$
    $R \leftarrow R \cup X_{actuel}$
    while $X_{actuel} \neq H_d$ *(ou* $X_{actuel} \neq B$*)* **do**
        mettreAJour$_{MOGWO}(X_\alpha, X_\beta, X_\delta)$
        foreach $X_i \in Pop$ **do**
            $X_i \leftarrow$ plusProcheChefVoisin(X_i)
        Evaluer(Pop)
        $NS \leftarrow$ solutionsNonDominées(Pop)
        mettreAJour$_{MOGWO}(archive, NS)$
        $X_\alpha, X_\beta, X_\delta \leftarrow$ meilleuresSolutions($archive$)
        $X_{actuel} \leftarrow X_\alpha$
        $R \leftarrow R \cup X_{actuel}$
    if $X_{actuel} = H_d$ **then**
        $R \leftarrow R \cup H_d$
    else
        $R \leftarrow R \cup B$

Le processus se répète tant que le chef actuel n'est pas la destination finale (H_d ou B). À chaque itération, OC met à jour les coordonnées des agents de recherche en fonction des coordonnées de X_α et de celles de deux autres leaders (X_β et X_δ), selon les équations fournies par MOGWO. Il met également à jour l'archive avec de nouvelles solutions non dominées et sélectionne les nouveaux chefs suivants à partir de cette archive.
Une fois que le dernier chef sélectionné est la destination finale (H_d ou B), cette destination est ajoutée à la route R comme dernier saut. Ce processus vise à achemine le message de manière efficace et optimisée en utilisant les chefs intermédiaires identifiés par l'algorithme MOGWO.
Les expérimentations menées pour tester et évaluer la contribution décrite dans cette section et les résultats correspondants sont présentés dans la section 5.2 du chapitre suivant. Cette contribution fait l'objet de notre publication [138].
Nous proposons une autre méthode de routage qui s'appuie sur une approche géographique, en utilisant une version améliorée de MOGWO appelée IMOGWO. Pour ce faire, nous commençons par introduire IMOGWO avant d'aborder l'autre méthode de routage. Les détails de notre contribution, IMOGWO, sont exposés dans la section suivante.

4.4 Improved MOGWO

La revue de l'état de l'art effectuée dans la section 3.2.6, résumée dans la synthèse 3.2.6, a souligné l'importance de continuer les efforts dans la recherche pour créer des méthodes de résolution qui offrent des performances élevées sur des problèmes standard et des cas réels, tout en étant capables de traiter un grand nombre d'objectifs. Dans ce contexte, GWO [3] et MOGWO [2], deux variantes mono-objectif et multi-objectifs de la méta-heuristique GWO 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 contribution décrite dans cette section consiste à proposer 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. L'IMOGWO introduit est représenté dans l'algorithme 9.
Afin d'expliquer l'algorithme 9, cette section considère qu'il est utilisé pour résoudre un problème d'optimisation de minimisation $P : min\ F(\overrightarrow{x}) = f_1(\overrightarrow{x}), f_2(\overrightarrow{x}), ..., f_o(\overrightarrow{x})$ où o est le nombre de

Algorithme 9 : Improved Multi-objective Gray Wolf Optimizer

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

initialiser(pop)
évaluer(pop)
$archive \leftarrow$ initialiserArchive()
$x_\alpha, x_\beta, x_\delta \leftarrow$ sélectionnerLeaders($archive$)
$it \leftarrow 1$
while $it \leq it_{max}$ **do**
  $w_\alpha, w_\beta, w_\delta \leftarrow$ calculerPoids($x_\alpha, x_\beta, x_\delta,$ $nombre\_bisections, valeursMinObjectifs, valeursMaxObjectifs$)
  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($x_i(t), x_{i\_gwo}$)
    $x_i(t+1) \leftarrow$ sélectionnerPosition($x_i(t), x_{i\_gwo}, x_{i\_dlh}$)
    $x_i(t) \leftarrow x_i(t+1)$
  mettreÀJourArchive($archive$)
  $x_\alpha, x_\beta, x_\delta \leftarrow$ sélectionnerLeaders($archive$)
  $it \leftarrow it + 1$
return $archive$

fonctions objectifs, $o \geq 2$. L'utilisation de IMOGWO pour résoudre ce problème de minimisation implique les étapes décrites par les sous-sections suivantes.

4.4.1 Initialisation

La phase d'initialisation correspond à :

— Initialiser l'espace de recherche : les n agents sont répartis aléatoirement dans l'espace de recherche. Une matrice Pop, ayant n lignes et d colonnes, est remplie avec les coordonnées des positions des n agents. Chaque agent est représenté par un vecteur $X_i(t) = \{x_{i1}, x_{i2}, ..., x_{id}\}$.

— Évaluer les valeurs de la fonction objectif de chaque X_i dans Pop.

— Initialiser l'archive (un espace de stockage) avec des solutions non dominées.

— Sélectionner les 3 meilleures solutions X_α, X_β et X_δ. Pour cela, le mécanisme proposé dans MOGWO est utilisé. Les solutions non dominées sont stockées dans l'archive selon une grille, dont la dimension correspond au nombre d'objectifs du problème d'optimisation. Chaque solution est placée dans un segment de la grille déterminé en fonction de ses valeurs objectives. Pour orienter la recherche vers des zones non explorées, les 3 meilleures solutions sont sélectionnées à partir du segment le moins encombré (le segment de la grille contenant le plus petit nombre de solutions).

4.4.2 Les différentes générations de IMOGWO

Tant que le nombre maximum d'itérations n'a pas été atteint, le calcul des poids et la mise à jour de la population de IMOGWO sont répétés. Chaque génération procède comme suit :

4.4.2.1 Le calcul des poids

Cette étape consiste à calculer 3 poids, w_alpha, w_beta et w_delta. Ces poids sont utilisés pour déterminer l'ordre d'importance entre X_α, X_β et X_δ. MOGWO utilise la même équation que GWO (algorithme 1) pour calculer les positions des loups, en garantissant que chaque élément suive les 3 meilleures solutions (X_α, X_β et X_δ). Ces solutions sont considérées avec la même importance et un coefficient égal à 1, mais elles sont classées en fonction de leur performance. Les coefficients w_alpha, w_beta et w_delta sont ensuite utilisés pour pondérer les contributions respectives de X_α, X_β et X_δ. Ainsi, l'équation utilisée pour mettre à jour les coordonnées des agents de recherche est exprimée comme suit (Équation 4.10) :

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

Les valeurs de w_alpha, w_beta et w_delta sont déterminées en suivant la méthode suivante. Le classement des trois meilleures solutions est considéré comme un problème d'optimisation distinct, noté $P_{classement}$. Ce problème partage les mêmes objectifs que le problème initial P : $f_1(\overrightarrow{x}), f_2(\overrightarrow{x}), ..., f_o(\overrightarrow{x})$. Étant donné que $P_{classement}$ se compose uniquement des solutions X_α, X_β et X_δ, sa dimension est fixée à 3. Par conséquent, on peut appliquer des méthodes de résolution exactes conçues pour des problèmes d'optimisation à petite échelle. Pour résoudre $P_{classement}$, on utilise la méthode de pondération, laquelle est la méthode exacte la plus directe pour résoudre des problèmes multi-objectifs à petite dimension. Cette méthode consiste à transformer le problème multi-objectifs initial en un problème mono-objectif en assignant un ordre de priorité aux objectifs et en associant à chacun un coefficient de pondération. Ensuite, les objectifs pondérés sont sommés pour obtenir une nouvelle fonction objectif. Ainsi, le problème initial est transformé en un problème mono-objectif dont l'objectif est de maximiser $\sum_{i\in 1..o} \lambda_i * f_i(x)$ sur $X_{classement}$, où λ_i représente le coefficient de pondération associé à la fonction objectif f_i. Cette transformation simplifie le classement des solutions possibles, car les différentes valeurs de l'objectif unique peuvent être directement ordonnées.
Les valeurs de w_alpha, w_beta et w_delta sont obtenues en appliquant l'Algorithme 10, qui prend en entrée les solutions X_α, X_β et X_δ et retourne leurs poids respectifs.
L'algorithme 10 utilise la technique de tri croisé [139] pour appliquer la méthode de pondération exacte. Dans un premier temps, une matrice carrée de dimensions $(3 * o) * (3 * o)$ est remplie

Algorithme 10 : Algorithme calculerPoids()

Input : x_alpha, x_beta, x_delta, obj_number, $bisection\_number$, $minValuesOfObjectives$, $maxValuesOfObjectives$
Output : w_alpha, w_beta, w_delta

$m \leftarrow$ une matrice carrée de 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

avec des valeurs entières attribuées aux fonctions objectifs pour chaque solution. Chaque ligne et chaque colonne de cette matrice correspondent à une fonction objectif pour une des solutions parmi X_α, X_β et X_δ. Les scores à inscrire dans chaque case sont calculés à partir des valeurs des fonctions objectifs associées à la ligne et à la colonne de la case. Ces scores sont calculés en utilisant l'algorithme 11, qui prend en entrée les valeurs des fonctions objectifs.

Algorithme 11 : Algorithme getValue()

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 ← 0
else
    if objective_index_1 <> objective_index_2 then
        score ← 0
    else
        d_value ←
          (maxValuesOfObjective − minValuesOfObjective)/bisection_number
        score ← ((sol_1.obj_value − sol_2.obj_value) Div d_value) + 1
return score
```

En premier lieu, l'algorithme 11 divise l'étendue des valeurs de la fonction objectif en un certain nombre de bisections, désignées par $bisection\_number$. Chaque sous-intervalle ainsi obtenu possède une largeur d_value. Le score renvoyé par cet algorithme correspond à la différence entre les valeurs de deux fonctions objectifs, divisée par d_value. Pour éviter un score nul, il est incrémenté de 1 dans tous les cas, sauf dans les cas spéciaux où les solutions correspondantes sont identiques ou lorsque la case concerne deux fonctions objectifs différentes.
Dans un second temps, les scores de chaque solution parmi X_α, X_β et X_δ sont sommés. Ensuite, $total\_sum$, la somme totale de tous les scores, est calculé. Le poids de chaque solution correspond alors au pourcentage de la somme de ses scores par rapport à $total\_sum$. La Figure 4.3 illustre la structure générale de la matrice utilisée et la procédure de calcul des poids.

4.4.2.2 La mise à jour de la population de IMOGWO

Pour chaque $X_i(t)$ dans Pop, la nouvelle position, $X_i(t+1)$, est calculée de la manière suivante :

— Utiliser les valeurs de w_alpha, w_beta et w_delta pour déterminer la première position candidate $X_{i-GWO}(t+1)$ en utilisant l'Équation 4.10.

— Calculer une seconde position candidate $X_{i-DLH}(t+1)$. Nous combinons ici le calcul d'une seconde position, utilisée dans IGWO, avec l'algorithme multi-objectifs. La méthode décrite

FIGURE 4.3 – Calcul des poids pour X_α, X_β et 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

dans le paragraphe (**a**) (sous-section 3.2.4.2) est employée pour déterminer cette seconde position candidate pour chaque loup à chaque itération.

— Sélectionner $X_i(t+1)$ parmi $X_i(t)$, $X_{i-GWO}(t+1)$ et $X_{i-DLH}(t+1)$ en appliquant l'algorithme 12. Les coordonnées de l'agent de recherche sont mises à jour uniquement si cette mise à jour optimise les objectifs. En premier lieu, l'algorithme 12 choisit entre X_{i-GWO} et X_{i-DLH} en sélectionnant la solution qui domine l'autre. Si aucune des solutions ne domine l'autre, X_{i-GWO} est choisi. Ensuite, il sélectionne entre la solution X_j précédemment choisie et la solution initiale $X_i(t)$. L'algorithme choisit la solution qui domine l'autre. Autrement, il retourne $X_i(t)$, maintenant ainsi la position de l'agent de recherche inchangée.

4.4.2.3 Mise à jour de l'archive

Cette étape consiste à mettre à jour l'archive avec les nouvelles solutions non dominées, conformément au processus utilisé par MOGWO. En d'autres termes, chaque solution non dominée trouvée est ajoutée à l'archive en suivant un processus qui inclut la vérification de la domination de cette solution par celles déjà présentes dans l'archive et la suppression des solutions dominées, le cas échéant. Si aucune solution dans l'archive ne domine la nouvelle solution, celle-ci est ajoutée à l'archive ; sinon, elle est rejetée.

4.4.2.4 Sélection des meilleures solutions

Cette étape sert à sélectionner les 3 nouvelles meilleures solutions (X_α, X_β et X_δ) parmi les solutions du segment le moins encombré dans l'archive.

Algorithme 12 : Algorithme sélectionnerPosition()

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

4.4.3 Les solutions finales

Les éléments qui restent dans l'archive après l'achèvement de toutes les générations de IMOGWO sont les solutions finales à retourner.
La performance de notre proposition IMOGWO a été évaluée en l'appliquant à la résolution des problèmes de référence DTLZ, ainsi qu'à un problème réel de modélisation de routage géographique multi-objectifs dans un réseau IoT. Les expérimentations menées et les résultats obtenus sont présentés dans les sections correspondantes du chapitre suivant : la section 5.3 pour la résolution des problèmes DTLZ et la section 5.5 pour le problème de routage géographique multi-objectifs.
Étant donné que le routage géographique requiert la connaissance des positions des objets, une solution de localisation devient nécessaire. Pour cela, nous avons envisagé de proposer une solution de localisation en intérieur dans les réseaux IoT. Notre idée est de bénéficier des techniques DL pour développer une approche efficace. Par conséquent, notre contribution, abordée en détail dans la section suivante, se concentre sur les techniques DL.

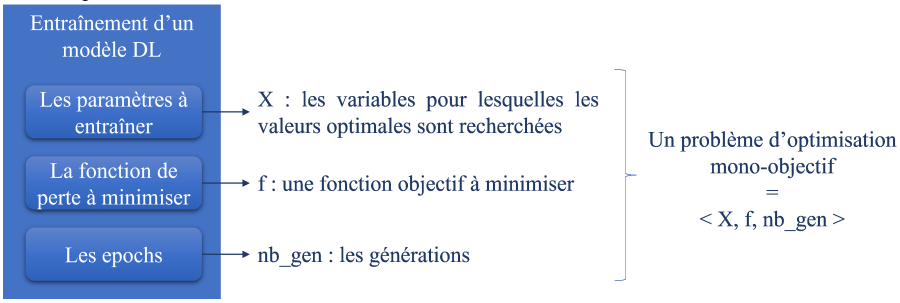
4.5 Optimisation des paramètres DL avec métaheuristiques

La revue de l'état de l'art effectuée dans la section 3.3.6, dont un résumé est présenté dans la synthèse 3.3.6, a mis en évidence l'importance de considérer l'optimisation des poids des mo-

dèles DL à l'aide de métaheuristiques. Cette approche permet de maximiser les avantages des modèles DL en accélérant leur convergence vers des valeurs optimales des poids et en améliorant leurs performances. De plus, il est essentiel que les optimiseurs développés soient conçus de façon qu'ils puissent être appliqués à différentes techniques de DL plutôt que d'être spécifiques à un seul type. Notre contribution, présentée dans cette section, s'inscrit dans ce contexte. Elle réside dans la modélisation de l'entraînement d'un modèle DL sous forme d'un problème d'optimisation mono-objectif, en établissant un lien entre l'ajustement des paramètres des couches de neurones et la résolution d'un problème d'optimisation à l'aide d'un algorithme métaheuristique. Cette approche vise à développer un optimiseur capable d'être utilisé pour l'entraînement de modèles DL, en hybridant ces deux paradigmes bénéfiques (optimisation et DL), tout en étant généralisable et applicable à plusieurs techniques de DL plutôt que spécifique à une seule.
Un modèle DL comprenant des couches denses est considéré, et l'apprentissage de leurs poids est optimisé au moyen d'algorithmes métaheuristiques. Comme évoqué dans la sous-section 3.3.3, lors de la phase d'entraînement, plusieurs itérations d'apprentissage sont effectuées afin de déterminer les valeurs des poids qui permettent de se rapprocher le plus à la fonction correcte permettant de mapper les entrées x aux sorties y. En d'autres termes, ces poids sont appris et améliorés au cours de l'entraînement. L'équation 3.19 simplifie et généralise l'équation de sortie d'une couche utilisée dans un réseau de neurones.
Les poids sont ajustés à chaque itération en utilisant un algorithme d'optimisation pour minimiser l'erreur de prédiction au fil des epochs. L'efficacité de l'optimiseur employé durant l'entraînement joue un rôle crucial dans l'augmentation de la vitesse de convergence du modèle DL et améliorer la qualité de ses prédictions. Dans le contexte des applications IoT, où une précision accrue et un traitement rapide sont essentiels, il est impératif d'améliorer les performances du modèle de localisation en utilisant un optimiseur plus performant. Pour relever ce défi, nous faisons appel à des métaheuristiques d'optimisation. En traitant l'apprentissage des poids du modèle DL comme un problème d'optimisation métaheuristique, nous pouvons utiliser des algorithmes spécifiques pour générer des solutions et mettre à jour ces poids de manière itérative.
En se basant sur cette conception de l'étape d'apprentissage, les poids d'un modèle DL sont optimisés dans cette étude en utilisant des algorithmes métaheuristiques. L'apprentissage des poids est ainsi formulé comme la résolution d'un problème d'optimisation mono-objectif. La Figure 4.4 illustre la transition de l'entraînement d'un modèle d'apprentissage profond vers un problème d'optimisation mono-objectif. Cette transition implique une redéfinition des composantes et des objectifs de l'entraînement d'un modèle DL en un problème d'optimisation avec un seul objectif à minimiser. Les composants sont transformés comme suit :

— Les paramètres à entraîner représentent les variables pour lesquelles on cherche les valeurs optimales. En DL, ces paramètres désignent les poids du réseau de neurones, ajustés lors de

FIGURE 4.4 – Conversion du processus d'entraînement d'un modèle de DL en un problème d'optimisation mono-objectif



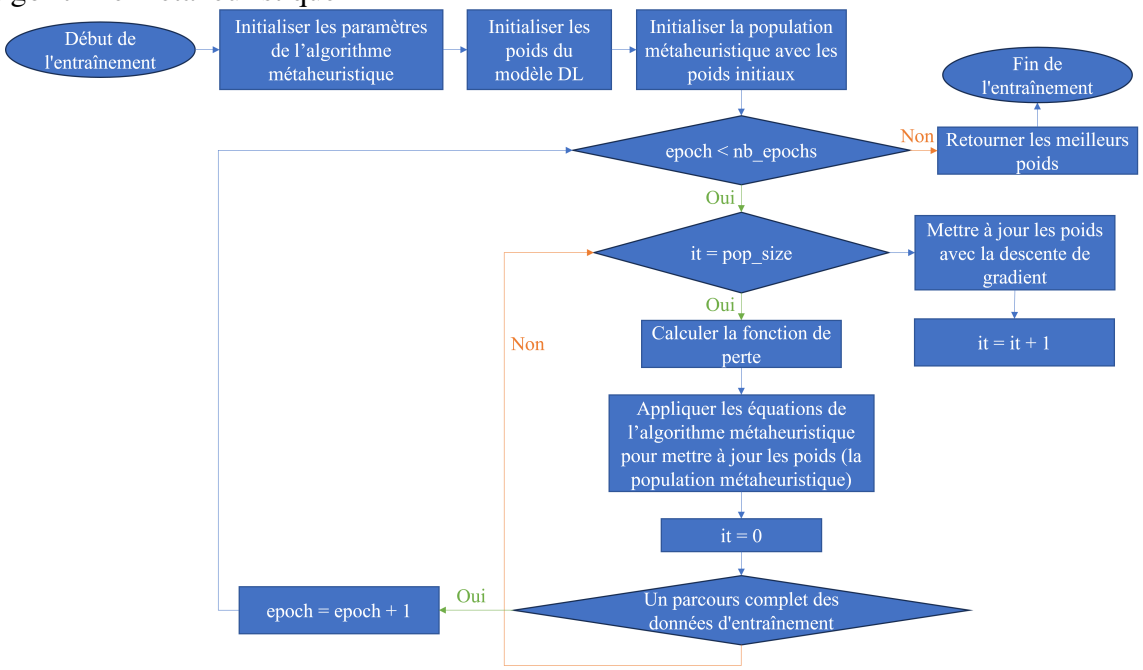
l'entraînement pour minimiser la fonction de perte. Dans le cadre de l'optimisation mono-objectif, ces paramètres deviennent les variables de décision X, visant à optimiser la fonction objectif.

— La fonction de perte (erreur de prédiction) à minimiser devient la fonction objectif. En DL, cette fonction mesure l'erreur entre les valeurs prédites et les valeurs réelles des variables. Dans le problème d'optimisation mono-objectif, la fonction objectif f est définie pour être minimisée en ajustant les variables de décision X, représentant les paramètres du modèle DL.

— Les epochs sont assimilées aux générations. En DL, les epochs correspondent au nombre de traitements de l'ensemble des données d'entraînement par le modèle. Dans l'optimisation mono-objectif, le nombre de générations représente les itérations ou générations de l'algorithme d'optimisation. À chaque itération, l'algorithme évalue la fonction objectif et ajuste les variables de décision pour obtenir de meilleures solutions, similairement à l'entraînement profond étalé sur plusieurs epochs pour améliorer les performances du modèle.

L'idée principale de l'approche consiste à considérer le processus d'entraînement du DL comme un problème d'optimisation, où l'objectif est de réduire au minimum la perte en ajustant les paramètres du modèle DL (variables de décision) à travers une série d'itérations ou de générations. Cette approche permet d'appliquer différentes techniques d'optimisation métaheuristiques mono-objectifs pour affiner le modèle et obtenir les meilleures performances possibles.
La représentation de l'optimisation des poids à l'aide d'une méthode métaheuristique est illustrée dans la Figure 4.5 et détaillée dans les Algorithmes 13, 14 et 15.
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

FIGURE 4.5 – Organigramme de l'entraînement des poids des couches de neurones en appliquant un algorithme métaheuristique



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.

L'algorithme 13 décrit en détail la phase d'entraînement du modèle développé. L'entraînement du modèle doit se faire par un ordinateur en lui donnant comme entrée les données d'apprentissage afin de fournir un modèle entraîné et capable de produire des résultats précis dans la phase de prédiction. Dans un premier temps, les poids du réseau de neurones sont initialisés avec des valeurs aléatoires, ainsi que les paramètres de l'algorithme métaheuristique à appliquer. Parmi ces paramètres, la taille de la population à considérer, désignée par pop_size, est notamment spécifié. Ensuite, un certain nombre d'epochs sont exécutées pour entraîner le modèle. Chaque epoch comprend un nombre d'itérations égal à la taille des données d'entrée divisée par la taille du lot à prendre en compte.

Les algorithmes 14 et 15 décrivent le processus effectué à chaque itération. Pendant les différentes epochs, la métaheuristique est appliquée après chaque itération de taille pop_size. En d'autres termes, la phase d'entraînement est divisée en étapes de longueur pop_size itérations. Au cours de la première étape, les valeurs de poids sont utilisées pour initialiser la population. À la fin de

Algorithme 13 : Entraînement des poids par un algorithme métaheuristique

Input : $input\_data$, pop_size, $nb\_generations$, nb_epochs
Output : $parameters$

```
1  weights ← poids aléatoires
2  metaheuristic_parameters ← initializeMetaheuristicParameters()
3  it ← 0
4  c_pop_write ← 0
5  c_pop_read ← 1
6  apply_metaheuristic_algo ← False
7  first_pop ← True
8  pop ← ∅
9  foreach epoch ∈ nb_epochs do
10     foreach step ∈ nb_iterations do
11         if apply_metaheuristic_algo est False then
12             constructOrUsePop(input_data,weights,pop, first_pop, pop_size,
                c_pop_write, c_pop_read, apply_metaheuristic_algo)
13         else
14             weights ← trainWithMetaheuristicAlgo(pop, nb_generations, it,
                metaheuristic_parameters)
```

cette étape, l'algorithme métaheuristique est utilisé pour mettre à jour les poids. Lors des étapes suivantes, les poids générés par l'étape précédente sont d'abord utilisés. À la fin de chaque étape, les poids sont optimisés en utilisant les équations de l'algorithme métaheuristique utilisé.
En résumé, à chaque itération, trois scénarios sont possibles : (i) les paramètres de DL sont initialisés de manière aléatoire et ajoutés à la population initiale de l'algorithme d'optimisation, (ii) les éléments de la population sont mis à jour en utilisant les équations de l'algorithme de métaheuristique, (iii) un élément de la population déjà modifié par l'algorithme d'optimisation est sélectionné pour être utilisé pendant l'entraînement.
L'algorithme 14 gère les situations (i) et (iii), tandis que l'algorithme 15 traite le cas (ii). Plus précisément, l'algorithme 14 concerne les itérations où l'algorithme d'optimisation métaheuristique n'est pas encore appliqué. Dans le premier cas, lors de la construction initiale de la population, les paramètres d'entraînement sont générés de manière aléatoire selon la méthode classique et ajoutés à la population. Ensuite, la taille de la population en cours de construction est vérifiée : si elle atteint sa taille maximale, deux actions sont entreprises :

— La variable booléenne $first\_pop$ est basculée sur $False$ pour éviter que ce processus ne se répète, car l'initialisation aléatoire de la population se produit uniquement une fois.

— La variable booléenne $apply\_metaheuristic\_algo$ est activée, indiquant que la population sera mise à jour en appliquant l'algorithme métaheuristique lors de la prochaine itération.

L'algorithme 14 traite le scénario (iii), où à la fois $apply\_metaheuristic\_algo$ et $first\_pop$ sont définis sur $False$. Dans cette situation, un élément de la population est sélectionné pour être utilisé lors de l'entraînement du modèle DL. Par la suite, le compteur c_pop_read, utilisé pour itérer à travers les éléments de la population, est incrémenté pour passer à l'élément suivant lors de la prochaine itération. Lorsque c_pop_read dépasse la taille de la population, cela signifie que tous les éléments de la population ont été utilisés et doivent être mis à jour avec l'algorithme métaheuristique lors de la prochaine itération. Ainsi, $apply\_metaheuristic\_algo$ est modifié à $True$.

L'algorithme 15 décrit le scénario (ii), où les équations de l'algorithme d'optimisation métaheuristique sont appliquées pour ajuster les éléments de la population (les paramètres) en vue des itérations ultérieures. Après cette adaptation, le premier élément de la population est extrait pour être utilisé comme paramètres du modèle DL lors de la prochaine itération. Cette procédure assure que la population évolue et s'adapte pour améliorer le processus d'optimisation. En outre, les paramètres de l'algorithme d'optimisation sont également actualisés avec les équations correspondantes afin de se préparer à la prochaine génération.

Algorithme 14 : Algorithme constructOrUsePop()

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

```
if first_pop is True then
    parameters ← getParametersWithTraditionalMethod()
    pop ← pop ∪ parameters
    c_pop_write ← c_pop_write + 1
    if c_pop_write = pop_size then
        first_pop ← False
        apply_metaheuristic_algo ← True
else
    parameters ← pop[c_pop_read]
    c_pop_read ← c_pop_read + 1
    if c_pop_read > pop_size then
        apply_metaheuristic_algo ← True
```

L'approche décrite dans cette section est mise en oeuvre pour concevoir une solution de localisation intérieure dans les réseaux IoT, dont les tests et l'évaluation sont présentés dans la section 5.4 du chapitre suivant. Cette solution de localisation repose sur la combinaison des techniques de DL et des métaheuristiques. La section suivante détaillera cette approche.

Algorithme 15 : Algorithme trainWithMetaheuristicAlgo()

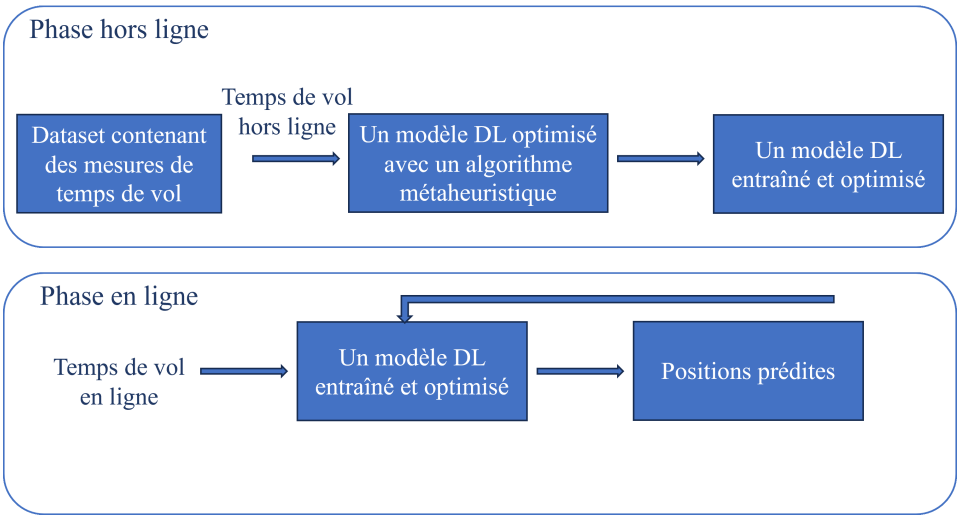
Input : pop, $nb\_generations$, $apply\_metaheuristic\_algo$, $metaheuristic\_parameters$
Output : new pop

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

4.6 Localisation en intérieur en hybridant des techniques DL et des métaheuristiques

Cette section est dédiée à l'explication de notre contribution à la localisation en intérieur dans un réseau IoT. L'approche proposée s'appuie sur notre contribution précédente, car elle repose sur un modèle de DL optimisé et entraîné à l'aide d'un algorithme d'optimisation métaheuristique. Le processus de localisation à développer comprend deux étapes (Figure 4.6) : une phase hors ligne et une phase en ligne.

FIGURE 4.6 – La localisation en appliquant un modèle DL optimisé et en utilisant des mesures UWB ToF



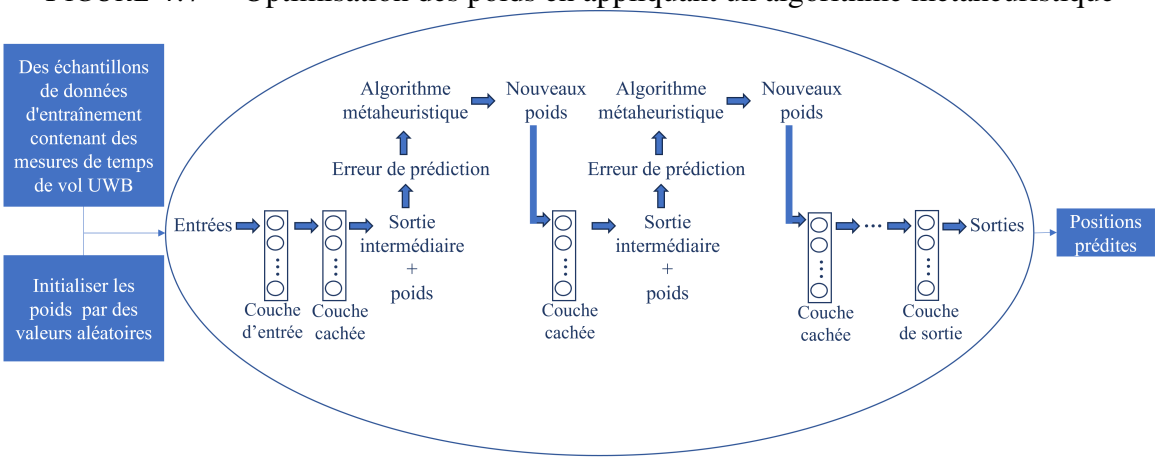
4.6.1 La phase hors ligne

Dans cette première étape, appelée la phase hors ligne (Figure 4.6(a)), nous procédons à l'entraînement du modèle de localisation à l'aide d'un dataset préparé préalablement, comprenant principalement des mesures de temps de vol UWB. Ces mesures sont prises à partir des signaux UWB provenant des communications entre des objets connectés dont les positions sont connues et des ancres fixes en profitant de la précision des mesures ToF permise par la technologie UWB. Les données préparées servent d'entrée au modèle DL que nous cherchons à entraîner. Dans ce cadre, la plateforme réelle LocURa4IoT [140] propose un dataset spécialement conçu pour élaborer et évaluer des méthodes de localisation en intérieur dans les environnements IoT, en se basant sur les mesures ToF et la technologie UWB. Ce dataset est disponible en ligne [141].
L'objectif principal de cette phase est d'apprendre au modèle l'association des mesures de temps de vol UWB aux coordonnées de l'objet mobile, afin qu'il puisse prédire avec précision la position de l'objet en fonction de ces mesures. En d'autres termes, nous voulons que le modèle apprenne la fonction permettant de mapper des mesures de temps de vol vers les coordonnées correspondantes d'un objet. Pour ce faire, nous utilisons un algorithme d'optimisation métaheuristique pour entraîner un modèle DL sur les données d'entrée. La Figure 4.7 illustre ce processus. Pendant le processus d'entraînement, le modèle ajuste ses poids internes pour minimiser une fonction de perte spécifique, qui mesure l'écart entre les positions prédites par le modèle et les positions réelles de l'objet mobile. Cet ajustement se fait en appliquant un algorithme métaheuristique car il guide le processus d'optimisation, orientant le modèle à s'améliorer au fil du temps en réduisant l'écart entre ses prédictions et les véritables positions.
De plus, pour évaluer la performance du modèle pendant l'entraînement et éviter le surajustement (overfitting), une partie des données est réservée à la validation. Cela permet de surveiller la capacité du modèle à généraliser ses prédictions à de nouvelles données non vues auparavant.
À la fin de cette phase, nous obtenons un modèle de localisation entraîné et optimisé, prêt à être utilisé dans la phase suivante du processus de localisation.

4.6.2 La phase en ligne

La phase en ligne (Figure 4.6(b)) constitue un aspect crucial de notre approche de localisation en intérieur dans un réseau IoT. Imaginons un scénario où un objet mobile, équipé de la technologie UWB, se déplace à l'intérieur du même espace utilisé dans la phase hors ligne. Les ancres fixes émettent des signaux UWB, permettant ainsi à l'objet mobile de déterminer sa position en mesurant les temps de vol de ces signaux entre les ancres et lui-même. Lorsque l'objet mobile souhaite se localiser, il commence par échanger des messages avec les ancres fixes par exemple par un protocole de type TWR [38] ou une version améliorée de celui-ci. Ces échanges de messages permettent

FIGURE 4.7 – Optimisation des poids en appliquant un algorithme métaheuristique



à l'objet d'obtenir des mesures de temps de vol UWB. Une fois que l'objet mobile a collecté les données nécessaires, il les transmet au modèle DL de localisation qui a été préalablement entraîné et optimisé lors de la phase hors ligne. Ce modèle utilise les mesures de temps de vol UWB en entrée pour estimer la position de l'objet mobile dans l'espace couvert par les ancres.

En résumé, l'approche proposée implique l'utilisation d'un modèle de localisation basé sur le DL, entraîné à l'aide d'un algorithme d'optimisation métaheuristique, pour estimer la position d'un objet mobile en se basant sur les mesures de temps de vol UWB collectées lors des échanges avec des ancres fixes. Cette approche permet à l'objet mobile de se localiser dans un environnement intérieur, en tirant parti des avantages de la technologie UWB et en combinant de manière synergique les deux paradigmes intéressants : l'optimisation métaheuristique et les techniques de DL. Les expérimentations visant à évaluer la contribution de localisation en intérieur décrite dans cette section, ainsi que les résultats correspondants, sont présentés dans la section 5.4 du chapitre suivant. Cette contribution fait l'objet de notre publication [142].

Les approches décrites dans les trois sections précédentes servent de base à l'élaboration d'une solution de routage géographique multi-objectifs au sein d'un réseau IoT. Cette approche est présentée en détail dans la section suivante.

4.7 Un routage géographique basé sur IMOGWO

Cette section se concentre sur notre contribution, détaillant une proposition de routage géographique hybride et multi-objectifs destinée à être appliquée dans les réseaux IoT à architecture maillée. Le routage géographique requiert une connaissance préalable des positions des objets connectés afin de diriger efficacement les messages à travers le réseau. Pour ce faire, notre méthode de routage exploite l'approche de localisation préalablement décrite dans la section 4.6, assurant

ainsi une connaissance précise des positions des objets. Par ailleurs, notre solution de routage est qualifiée de multi-objectifs, car elle applique notre proposition IMOGWO, présentée dans la section 4.4, pour optimiser le processus de routage en tenant compte des objectifs décrits dans la section 4.2.2 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 de cette approche est de garantir un acheminement efficace des messages tout en répondant aux exigences des réseaux IoT. Dans notre approche, la construction des routes est distribuée entre les objets du réseau IoT sans nécessiter d'ordinateur central pour gérer le routage. Cette décentralisation correspond à l'idée fondamentale de l'Edge Computing, où le traitement des données et les décisions sont effectués au plus près des sources de données pour réduire la latence, économiser la bande passante et améliorer la résilience du système. Cette section présente le modèle de réseau à considérer ainsi que les hypothèses à prendre en compte. De plus, elle détaille les algorithmes qui décrivent le processus du routage à développer.

4.7.1 Le modèle de réseau

Le modèle de réseau IoT que nous envisageons est caractérisé par une topologie maillée, où chaque noeud est connecté directement à tous les noeuds qui se trouvent dans sa portée de communication. La technologie de communication utilisée est l'UWB. Ce réseau comprend une diversité d'objets avec des capacités énergétiques variables, comprenant à la fois des objets fixes et des objets mobiles. Pour garantir une localisation précise des objets mobiles, des ancres fixes sont déployées dans le réseau. Dans ce contexte de réseau IoT à topologie maillée, plusieurs défis se présentent. La dynamique de l'environnement peut entraîner des changements imprévus dans la topologie du réseau, ce qui rend difficile la sélection des chemins optimaux pour la transmission des données.

4.7.2 Les hypothèses

Les hypothèses à prendre en compte pendant le processus de routage sont :

— Le réseau est composé d'objets hétérogènes, présentant des capacités énergétiques variées, incluant à la fois des objets fixes et des objets mobiles.

— Le routage est multi-sauts : la construction d'une route est distribuée et partagée entre les objets parcourus, où chaque objet calcule le prochain saut. En d'autres termes, le processus de routage ne nécessite pas d'ordinateur central pour gérer le routage.

— Les fonctions objectifs du routage concernent un saut et sont évaluées à chaque saut.

FIGURE 4.8 – Un exemple explicatif d'une table de voisinage pour un objet dans le réseau

Table de voisinage pour Object 20

id	x	y	Energie	Confiance	Voisins id	x	y	Energie	Confiance
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
					...				
...									

— En cas de problèmes avec un objet pendant l'acheminement, tel qu'une impossibilité d'avancer, il est nécessaire de revenir en arrière (d'un seul saut) et de recommencer à partir de l'objet précédent dans la route parcourue, en reprenant la progression positive dès que possible.

— Les objets commencent avec un même degré de confiance qui change au fur et à mesure des expériences précédentes avec les objets connus (voisins).

— Pour acheminer les messages, chaque objet doit connaître sa propre position dans le réseau, l'emplacement de la destination finale, ainsi que des informations concernant ses voisins à deux sauts (position, énergie résiduelle, degré de confiance).

— Chaque objet mobile diffuse sa position à ses voisins dès qu'il se déplace à une distance de référence spécifiée.

— Chaque objet diffuse régulièrement des informations à ses voisins, notamment sa position, les degrés de confiance, l'énergie résiduelle et le nombre de voisins des objets connus.

— Chaque objet écoute ses voisins pour construire et mettre à jour une table de voisinage à 1 et 2 sauts, comprenant la position, l'énergie résiduelle et le degré de confiance. La figure 4.8 présente un exemple explicatif d'une table de voisinage pour un objet dans le réseau.

4.7.3 Le processus du routage

Le routage que nous proposons est de nature géographique, réactive et multi-objectifs. Il repose essentiellement sur les positions des objets connectés dans le réseau pour acheminer les messages. Notre approche de routage prend en considération les objectifs décrits dans la section 4.2.2 afin d'optimiser le cheminement des messages et garantir des communications efficaces. 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.
Le processus de routage commence par l'algorithme 16. Il débute par la vérification de la présence de l'objet destination finale parmi les voisins accessibles à 1 ou 2 sauts dans la table de voisinage à 2 sauts de la source initiale. Si tel est le cas, le routage est proactif, ce qui signifie que la route est déjà connue : l'objet source envoie le message directement à l'objet destination finale ou via un objet voisin. En revanche, si l'objet destination finale n'est pas répertorié dans la table de voisinage, le routage géographique réactif est activé (algorithme 17) afin de déterminer la meilleure route pour acheminer le message.

Algorithme 16 : Algorithme de routage basé sur IMOGWO

Input : m, o_s, o_d, $DLModel$
Output : acheminer m à o_d

```
if o_d ∈ T_v(o_s) then
    if o_d ∈ V(o_s) then
        o_s envoit m à o_d
    if o_d ∈ V2S(o_s) then
        o_s envoit m à o_v(∈ V(o_s))
        o_v envoit m à o
else
    R ← lookForPathToDestination(o_s, o_d, m, DLModel)
```

L'algorithme 17 vise à déterminer une route optimisée pour acheminer le message m de l'objet source initial o_s vers l'objet destination finale o_d en appliquant l'algorithme IMOGWO. Le processus commence par la recherche de la position de l'objet destination finale o_d en diffusant des requêtes dans le réseau à partir des voisins de l'objet courant. Si l'objet source o_s est mobile, sa position est calculée en utilisant le modèle DL de localisation, $DLModel$, après l'échange des messages avec les ancres fixes dédiées pour la localisation. Ensuite, l'algorithme initialise, Pop, la population de IMOGWO en incluant les voisins de l'objet courant o_s, puis évalue chaque membre de la population en calculant leur fitness pour chaque fonction objectif et en utilisant les positions de o_s et o_d.
Après avoir identifié les solutions non dominées NDS, l'algorithme initialise une archive et sélectionne les meilleurs leaders parmi ces solutions en appliquant les mécanismes de l'algorithme IMOGWO. Ensuite, l'objet source o_s envoie le message m et des informations sur les leaders sélectionnés et la population Pop à X_α, le meilleur parmi les leaders choisis. Par conséquent, X_α devient l'objet courant o_c (où le message arrive pendant l'acheminement).

Ensuite, o_c est mis à jour itérativement jusqu'à ce qu'il atteigne l'objet destination finale o_d. À chaque itération, si l'objet courant o_c est mobile, sa position est calculée en utilisant le modèle de localisation $DLModel$, puis les coordonnées des membres de la population Pop sont mises à jour en utilisant les équations de l'algorithme IMOGWO (algorithme 18). À la suite, pour chaque élément X_i de la population Pop, une méthode de correspondance est utilisée afin de faire correspondre les coordonnées calculées de X_i avec les coordonnées d'un objet réel parmi l'ensemble des voisins $V(o_c)$ de l'objet courant o_c (algorithme 21). Ensuite, pour chaque élément de la population, le fitness est calculé pour chaque fonction objectif. Après avoir effectué ces calculs, l'algorithme recherche les solutions non dominées parmi les éléments de la population Pop et met à jour l'archive avec ces solutions selon l'algorithme IMOGWO (algorithme 22). Enfin, les nouvelles meilleures solutions sont sélectionnées parmi les éléments de l'archive en utilisant les mécanismes de l'algorithme IMOGWO, et les leaders sélectionnés sont exclus de l'archive pour la prochaine itération.
Enfin, l'objet courant o_c envoie le message au prochain leader X_α et décide de suivre la route déterminée par l'algorithme ou de revenir en arrière dans la route parcourue selon la disponibilité de voisins permettant un avancement positif vers l'objet destination final.
Une fois que o_c atteint o_d, cela indique que la destination finale a été atteinte et que l'acheminement du message est achevé.
L'algorithme 18 permet de mettre à jour les coordonnées de l'objet X_i en utilisant les équations de l'algorithme IMOGWO. L'algorithme prend en entrée l'objet X_i ainsi que les leaders X_α, X_β et X_δ. Il commence par calculer X_{i-GWO} en utilisant GWO avec X_i, X_α, X_β et X_δ comme entrées. Ensuite, il calcule X_{i-DLH} en utilisant DLH avec X_{i-GWO} comme entrée. Le calcul de ces deux positions est présenté dans les algorithmes 19 et 20. Ensuite, l'algorithme compare les solutions X_{i-GWO} et X_{i-DLH} pour déterminer laquelle domine l'autre. Si X_{i-GWO} domine X_{i-DLH}, alors $X_{i-(t+1)}$ est mis à jour avec X_{i-GWO}. Si X_{i-DLH} domine X_{i-GWO}, alors $X_{i-(t+1)}$ est mis à jour avec X_{i-DLH}. Si aucune des deux ne domine l'autre, alors $X_{i-(t+1)}$ est choisi aléatoirement entre X_{i-GWO} et X_{i-DLH}. Enfin, si la solution $X_{i-(t+1)}$ domine la solution actuelle X_i, alors les coordonnées de X_i sont mises à jour avec celles de $X_{i-(t+1)}$.
Les deux algorithmes 19 et 20 modélisent le calcul des nouvelles positions candidates de X_i selon les équations exposées dans la section 4.4.
L'algorithme 21 a pour objectif de mapper les coordonnées d'un objet X_i, parmi les éléments de Pop, vers les coordonnées d'un objet réel, parmi les voisins de l'objet courant o_c. L'algorithme commence par initialiser une liste vide $positions$ pour stocker les positions des voisins de l'objet courant. Ensuite, pour chaque voisin v_j dans l'ensemble $V(o_c)$, l'algorithme ajoute sa position à la liste $positions$. Après avoir collecté toutes les positions des voisins, l'algorithme détermine le voisin le plus proche $V_{nearest}$ parmi les positions enregistrées. Enfin, il retourne la position du voisin le plus proche, qui servira de nouvelle position pour l'objet X_i.

Algorithme 17 : Algorithme lookForPathToDestination()

Input : $o_s, o_d, m, DLModel$
Output : La route de o_s jusqu'à o_d

$(x_d, y_d) \leftarrow$ searchPosition(o_d)
if *isMobile(o_s)* **then**
  $(x_s, y_s) \leftarrow$ computePosition(o_s, $DLModel$)
$o_c \leftarrow o_s$
$Pop \leftarrow \varnothing$
foreach $v_i \in V(o_c)$ **do**
  $Pop \leftarrow Pop \cup \{v_i\}$
foreach $X_i \in Pop$ **do**
  foreach $f_i \in$ *objectiveFunctions* **do**
    Calculer le fitness de X_i pour f_i
$NDS \leftarrow$ findNonDominatedSolutions(Pop)
$archive \leftarrow$ initializeArchive(NDS)
$\{X_\alpha, X_\beta, X_\delta\} \leftarrow$ selectLeaders($archive$)
exclure $\{X_\alpha, X_\beta, X_\delta\}$ de $archive$
o_s envoit m, X_β, X_δ et Pop à X_α
$o_c \leftarrow X_\alpha$
while $o_c \neq o_d$ **do**
  if *isMobile(o_c)* **then**
    $(x_c, y_c) \leftarrow$ computePosition(o_c, $DLModel$)
  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(o_c,X_i,$V(o_c)$)
  foreach $X_i \in Pop$ **do**
    foreach $f_i \in$ *objectiveFunctions* **do**
      Calculer le fitness de X_i pour f_i
  $NDS \leftarrow$ findNonDominatedSolutions(Pop)
  updateArchive($archive$, NDS)
  $\{X_\alpha, X_\beta, X_\delta\} \leftarrow$ selectLeaders($archive$)
  exclure $\{X_\alpha, X_\beta, X_\delta\}$ de $archive$
  o_c envoit m, X_β, X_δ et Pop à X_α
  if *canEnablePositiveProgress($V(o_c)$)* **then**
    $o_c \leftarrow X_\alpha$
  else
    $o_c \leftarrow$ previous o_c

Algorithme 18 : Algorithme updateCoordinates()

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**
  $X_i \leftarrow X_{i-(t+1)}$

Algorithme 19 : Algorithme calculateXGWO()

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}

Algorithme 20 : Algorithme calculateXDLH()

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

$R_i \leftarrow ||X_i - X_{i-GWO}||$
$N_i \leftarrow \{X_j | distance\_euclidienne(X_i, X_j) \leq R_i, X_j \in Pop\}$
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}

Algorithme 21 : Algorithme mapFromComputedCoordinatesToRealPosition()

Input : o_c,X_i,$V(o_c)$
Output : nouveau X_i

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

L'algorithme 22 met à jour l'archive avec les nouvelles solutions non dominées NDS. Il prend en entrée l'archive actuelle $archive$ et l'ensemble des nouvelles solutions non dominées NDS. Pour chaque solution Sol_i dans NDS, l'algorithme vérifie si elle peut être ajoutée à l'archive en respectant les règles de domination. Si l'archive est vide, la solution Sol_i est ajoutée directement à l'archive. Sinon, pour chaque solution S déjà présente dans l'archive, l'algorithme vérifie si Sol_i la domine. Si Sol_i domine S, alors S est exclue de l'archive. Si l'archive est pleine après l'ajout de Sol_i, l'algorithme réorganise la segmentation de la grille et exclut une solution du segment le plus encombré. Enfin, la nouvelle solution Sol_i est ajoutée à l'archive mise à jour.

Algorithme 22 : Algorithme updateArchive()

Input : NDS
Output : archive mise à jour

foreach $Sol_i \in NDS$ **do**
  if $archive$ *est vide* **then**
    $archive \leftarrow archive \cup \{Sol_i\}$
  else
    if $\nexists S \in archive | S \succ Sol_i$ **then**
      foreach $S \in archive$ **do**
        if $Sol_i \succ S$ **then**
          $archive \leftarrow archive \setminus S$
      if $archive$ *est pleine* **then**
        Réorganiser la segmentation de la grille
        Exclure une solution du segment le plus encombré
      $archive \leftarrow archive \cup \{Sol_i\}$

La section 5.5 du chapitre suivant présente les expérimentations ainsi que les résultats d'évaluation de notre proposition de routage géographique détaillée dans cette section.

4.8 Conclusion

En conclusion, ce chapitre a exposé une série de contributions théoriques que nous avons proposées. Nous avons d'abord introduit une modélisation mathématique du routage dans les réseaux IoT, suivie d'une proposition d'amélioration du MOGWO que nous avons détaillée. En outre, nous avons proposé l'optimisation des paramètres des modèles DL à l'aide de métaheuristiques, ainsi que de son application pour élaborer une approche la localisation en intérieur dans les réseaux IoT. Enfin, nous avons consacré une section à décrire notre approche de routage géographique hybride et multi-objectifs, destinée à être déployée dans les réseaux IoT à architecture maillée. Cette approche s'inscrit dans le cadre de l'Edge computing puisque la prise de décision de routage est distribué entre les objets du réseau. En d'autres termes, le processus de routage ne nécessite pas un ordinateur central qui calcule les routes pour acheminer les données dans le réseau. Le chapitre suivant se concentrera sur les résultats numériques des tests et des expérimentations menées pour valider et évaluer les contributions théoriques présentées dans ce chapitre.

Chapitre 5

Résultats numériques, simulations et expérimentations

Sommaire

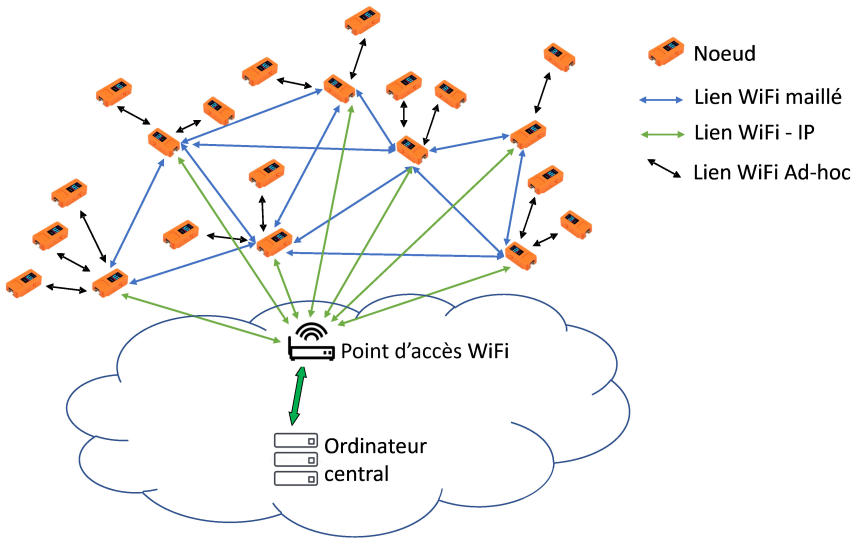
5.1 Introduction

Le chapitre précédent a initié nos contributions théoriques en présentant la modélisation mathématique du routage dans les réseaux IoT proposée, accompagnée de propositions novatrices. Nous avons introduit un algorithme de routage réactif et multi-objectifs basé sur l'optimiseur métaheuristique MOGWO. De plus, nous avons introduit un algorithme, Improved MOGWO, une version améliorée de MOGWO, ainsi qu'une méthode d'optimisation des paramètres des modèles DL en utilisant des métaheuristiques. En outre, nous avons proposé une approche de localisation en intérieur combinant les techniques DL et les métaheuristiques. En tirant parti de ces approches, nous avons développé un modèle de routage géographique hybride et multi-objectifs conçu pour les réseaux IoT dotés d'une architecture maillée. Ce chapitre est dédié à la présentation et à l'analyse des expérimentations menées pour évaluer nos contributions. La première section détaille les expériences menées pour évaluer notre proposition de routage réactif et multi-objectifs basé sur MOGWO. Nous analysons les résultats obtenus, mettant en évidence les performances de notre approche. La deuxième section se concentre sur les expériences visant à tester la résolution des problèmes de référence DTLZ avec notre proposition IMOGWO. Nous évaluons théoriquement ses performances en comparant les résultats obtenus avec ceux d'autres algorithmes d'optimisation largement utilisés. La troisième section décrit la configuration expérimentale des tests réalisés pour évaluer l'efficacité de nos contributions relatives à l'optimisation des poids des modèles DL par des algorithmes métaheuristiques, ainsi que le développement d'un modèle DL de localisation en intérieur basé sur les mesures UWB ToF. De plus, elle présente les résultats obtenus lors de l'évaluation de ces approches et les compare à ceux d'autres solutions de localisation en intérieur. Enfin, la dernière section présente et analyse les résultats des expérimentations sur notre proposition de routage géographique et hybride. Nous évaluons les performances de cette solution, les comparant à celles d'autres solutions de routage existantes, et testons l'efficacité de l'algorithme IMOGWO en l'appliquant à un problème de routage réel, en dehors des problèmes de référence DTLZ.

5.2 Résultats expérimentaux du routage basé sur MOGWO

Cette section est dédiée à présenter les expérimentations menées pour évaluer notre contribution de routage réactif et multi-objectifs basé sur l'algorithme d'optimisation métaheuristique MOGWO, comme détaillé dans la section 4.3, ainsi qu'à analyser les résultats obtenus.

FIGURE 5.1 – Architecture du réseau proposée



5.2.1 Les détails des expérimentations

Le tableau 5.21 présente les paramètres des expérimentations menées pour évaluer l'algorithme de routage proposé. Ces expérimentations ont été réalisées sur un réseau IoT contenant des objets connectés récents Wi-Fi/BLE 2.4 GHz appelés M5StickC [143]. Dans la configuration expérimentale, 22 objets M5StickC 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. Le noeud M5StickC est équipé d'un émetteur infrarouge, d'un microphone, d'un écran LCD et d'une mémoire interne de 4 Mo. Pour exploiter ces noeuds M5StickC, nous avons utilisé la bibliothèque Arduino Painless Mesh [144] qui, contrairement à la bibliothèque ESP-NOW [145] utilisée dans [146], permet l'accès à une série de fonctions d'émission et de réception de trames sans fil, dans une topologie pouvant se baser sur une hiérarchie arborescente. Aussi, ESP-NOW exige que tous les noeuds doivent être à portée radio de tous les autres, car il n'y a pas de routage multi-saut prévu [146]. Les paramètres expérimentaux comprennent le nombre moyen d'exécutions, fixé à 25, ainsi que la technologie de communication utilisée, qui est le Wi-Fi. La portée de transmission/détection est d'environ 27 mètres en moyenne. La puissance de transmission utilisée pour les expérimentations est de 100 mW. Un scénario d'échange de trames entre des noeuds aléatoires du réseau est utilisé pour tester la qualité des liens et du routage.
La Figure 5.1 représente l'architecture du réseau proposée.

TABLEAU 5.1 – Les paramètres des expérimentations

Nombre moyen d'exécutions	25
Surface	300 * 300 m^2
Nombre d'objets M5StickC	22 objets fixes et 1 objet mobile
Technologie de communication	Wi-Fi
Portée de transmission/Détection	27 m (valeur moyenne)
RSSI	Variable (initialisé à 120 milliwatts)
Frame Error Rate(FER)	Variable (initialisé à 0.01)
Puissance de transmission	100 mW

5.2.2 Les algorithmes de routage comparés

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]. Le tableau 5.2 montre les paramètres utilisés pour les deux métaheuristiques multi-objectifs appliqués, MOGWO et NSGA-III.

TABLEAU 5.2 – Les paramètres de MOGWO et NSGA-III

Paramètre			**Valeur**
Nombre d'exécutions			25, en utilisant des populations initiales distinctes
Nombre de générations			10 à 650
Taille de la population			300
Nombre d'objectifs			3 à 5
MOGWO	Taille de l'archive		50
	alpha		0.1
	beta		4
	gamma		2
NSGA-III	Mutation	Opérateur	Swap Mutation
		indice de distribution	45
		probabilité	0.0025
	Cross-over	Opérateur	"Sequential Constructive" (SCX)
		indice de distribution	25
		probabilité	0.9

5.2.3 Les critères d'évaluation

Afin d'évaluer les comportements des optimiseurs multi-objectifs appliqués pour développer notre approche de routage proposée, les critères d'évaluation suivants sont utilisés :

— L'hypervolume (voir section 3.2.5) permettant d'évaluer la convergence et la distribution des solutions générées par les deux algorithmes d'optimisation comparés ;

— La durée de vie moyenne du réseau exprimant 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 ;

— Le nombre moyen de voisins des objets dans le réseau pour évaluer la capacité des deux solutions de routage comparés à maintenir une bonne répartition des objets dans l'environnement IoT étudié ;

— Le délai de transmission entre la source initiale et la destination finale ;

— La valeur RSSI moyenne qui représente le signal reçu et sa qualité.

5.2.4 Résultats et discussion

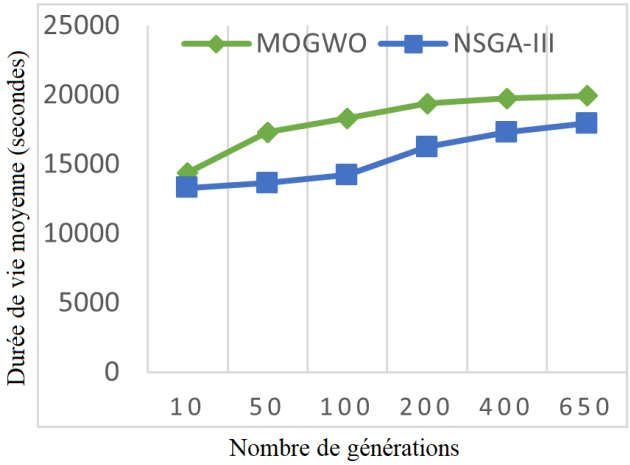
Le tableau 5.3 présente les résultats en termes de l'indicateur Hypervolume obtenus par les deux algorithmes de routage comparés. Trois scénarios de tests ont été exécutés, variant le nombre de générations et le nombre d'objectifs considérés. Dans le premier scénario, avec 350 générations et un seul objectif, MOGWO a dépassé NSGA-III en termes de meilleure et pire performance en Hypervolume, avec respectivement 0.921594 et 0.901168. Cependant, NSGA-III a surpassé MOGWO en termes de valeur moyenne. Dans le deuxième scénario, avec 450 générations et deux objectifs, MOGWO a également surpassé NSGA-III pour les cas de meilleure et pire performance, avec respectivement 0.982056 et 0.979087. Cependant, pour la valeur moyenne, NSGA-III a présenté une légère supériorité. Dans le troisième scénario avec 650 générations et trois objectifs, MOGWO a maintenu sa supériorité en obtenant une meilleure performance avec un Hypervolume de 0.973267, tandis que NSGA-III a obtenu 0.921185. De plus, MOGWO a également surpassé NSGA-III en termes de moyenne valeur, bien que les performances dans le pire cas sont quasiment identiques. En conclusion, les résultats indiquent que MOGWO présente généralement de meilleures performances en termes d'Hypervolume par rapport à NSGA-III dans différents scénarios de tests.

TABLEAU 5.3 – 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

La figure 5.2 compare les deux algorithmes de routage en termes de la durée de vie moyenne du réseau obtenus pour les différents nombres de générations exécutées. Globalement, nous observons une augmentation de la durée de vie moyenne du réseau avec le nombre croissant de générations

FIGURE 5.2 – La durée de vie moyenne du réseau pour le routage modélisé avec MOGWO et NSGA-III



pour les deux algorithmes. Cependant, des différences significatives sont remarquées entre les deux approches. Pour le routage avec MOGWO, une progression relativement constante de la durée de vie moyenne du réseau est constatée à mesure que le nombre de générations augmente, illustrant ainsi la capacité de cet algorithme à maintenir une amélioration progressive des performances énergétiques du réseau. En revanche, pour le routage avec NSGA-III, bien que la durée de vie moyenne du réseau augmente également avec le nombre de générations, les valeurs sont généralement inférieures à celles obtenues avec MOGWO. Cela montre que, malgré une optimisation croissante à mesure que le nombre de générations augmente, NSGA-III ne parvient pas à égaler MOGWO en termes d'efficacité énergétique et de maintien de l'équilibre de charge entre les objets du réseau. En résumé, 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.

La figure 5.3 compare les deux algorithmes en termes du nombre moyen de voisins des objets dans le réseau. Les résultats illustrés indiquent qu'en moyenne, le nombre de voisins des objets est plus élevé lorsque le routage est effectué avec MOGWO par rapport à NSGA-III pour chaque nombre de générations considéré, ce qui indique que le routage basé sur MOGWO tend à maintenir une meilleure répartition des objets dans l'environnement IoT étudié, ce qui pourrait contribuer à une meilleure efficacité et une réduction des congestions réseau.

D'autre part, la figure 5.4 présente les résulats d'évaluation de deux algorithmes comparés en termes de délai de transmission entre la source initiale et la destination finale. Ainsi, l'analyse des résultats montre qu'en moyenne, le délai de transmission est plus court lorsque le routage est effectué avec MOGWO 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

FIGURE 5.3 – Le nombre moyen de voisins pour le routage modélisé avec MOGWO et NSGA-III

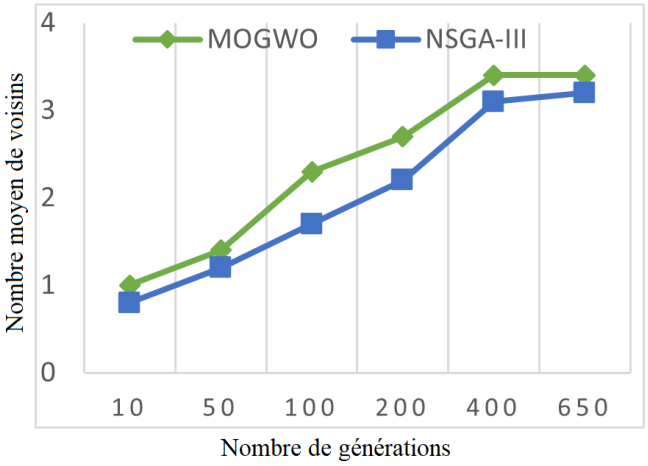
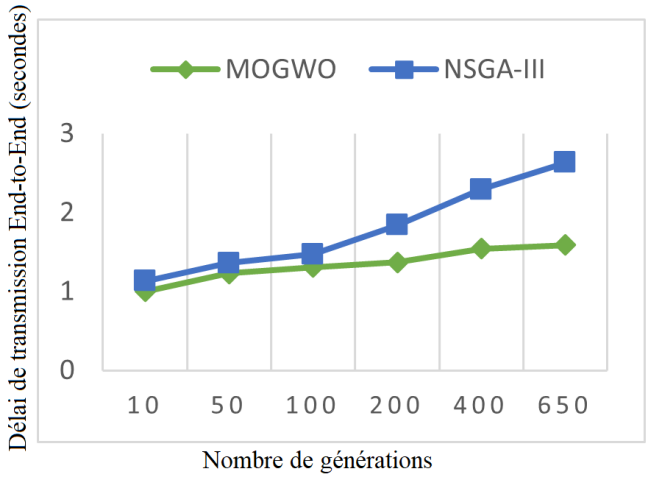


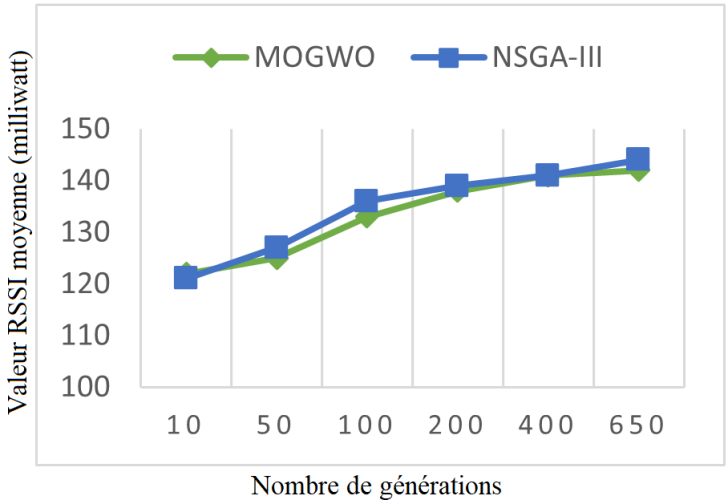
FIGURE 5.4 – Délai de transmission end-to-end pour le routage modélisé avec MOGWO et NSGA-III



développer l'algorithme de routage proposé, ce qui peut contribuer à une meilleure réactivité du réseau et à une livraison plus rapide des données.

En outre, la figure 5.5 présente les valeurs RSSI moyennes obtenues par les deux algorithmes de routage évalués pour les différents nombres de générations considérés. Ces valeurs ont été converties en milliwatts. L'analyse des résultats présentés montre que, en moyenne, la valeur RSSI est légèrement plus élevée lorsque le routage est effectué avec MOGWO par rapport à NSGA-III pour chaque nombre de générations considéré. Cela montre que MOGWO tend à offrir une meilleure qualité de signal en termes de force de réception par rapport à NSGA-III, assurant ainsi des communications plus fiables et plus stables dans le réseau IoT étudié.

FIGURE 5.5 – La valeur RSSI moyenne pour le routage modélisé avec MOGWO et NSGA-III



5.2.5 Synthèse

En effet, l'analyse comparative entre les deux algorithmes de routage multi-objectifs pour les réseaux IoT à topologie hybride, qui sont basés sur MOGWO et NSGA-III, révèle que MOGWO surpasse généralement NSGA-III sur plusieurs critères. En fait, 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. En outre, 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].

5.3 Résultats numériques de IMOGWO sur les problèmes de test DTLZ

Dans le but d'évaluer théoriquement les performances de l'algorithme IMOGWO proposé, des expériences ont été menées pour tester la résolution des problèmes de référence DTLZ. Les résultats obtenus sont comparés à ceux obtenus par des algorithmes d'optimisation récents et largement utilisés. Pour ce faire, nous avons utilisé jMetal[147], un framework écrit en Java dédié à l'optimisation multi-objectifs, pour calculer les indicateurs de performance. Les expériences ont été exécutées sur les serveurs de calculs informatiques de l'Institut de Recherche en Informatique de Toulouse (IRIT) [148], des plateformes permettant d'accéder à des ressources informatiques à haute performance.

Cette section présente en détail les expériences réalisées et analyse les résultats obtenus. Des tests statistiques ont également été appliqués pour comparer l'efficacité de l'optimiseur proposé à celle des autres optimiseurs étudiés.

5.3.1 Détails des tests

Cette section présente les algorithmes utilisés et les problèmes de test standard ainsi que les indicateurs de performance utilisés pour comparer et évaluer numériquement les algorithmes étudiés.

5.3.1.1 Algorithmes de comparaison et paramètres de configuration

L'algorithme proposé (IMOGWO) a été comparé au 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 paramètres initiaux utilisés par les algorithmes étudiés sont présentés dans le tableau 5.4. Les expériences ont été réalisées avec le framework jMetal [147] qui utilise des méthodes de réglage automatique pour configurer les algorithmes et déterminer les paramètres typiques. Les paramètres initiaux de DMOPSO, MOEA/D-IEps, MOEA/DD et ESPEA sont donc les mêmes que ceux utilisés dans jMetal. De plus, les paramètres de MOGWO correspondent à ceux mentionnés dans l'étude qui l'a proposé, [2], où la taille de la population et le nombre de générations étaient fixés à 100 et 3000 respectivement. Pour garantir une comparaison équitable entre les optimiseurs étudiés, ces mêmes valeurs ont été utilisées pour les autres algorithmes. Ainsi, une population de 100 individus a été utilisée pour tous les tests, avec des éléments générés aléatoirement respectant les domaines des variables de chaque problème. Le nombre de générations pour chaque algorithme était de 3000. Tous les algorithmes ont été exécutés 30 fois sur les problèmes de test.

TABLEAU 5.4 – Paramètres initiaux utilisés par les algorithmes étudiés

Algorithme	IMOGWO	MOGWO	DMOPSO	MOEA/D-IEps	MOEA/DD	ESPEA
Paramètres	Nombre de bi-sections = 10	Nombre de bi-sections = 10	r1, r2 : nombres réels aléatoires entre 0.0 et 0.1 Paramètres cognitifs, sociaux : nombres réels aléatoires entre 1.5 et 2.5 Poids d'inertie : 0.4	Taille du voisinage : 30 Probabilité de sélection de voisinage : 0.9	Taille du voisinage : 20 Probabilité de sélection de voisinage : 0.9	Probabilité de croisement = 0.9 Indice de distribution du croisement = 20.0
	Taille de la population = 100 Nombre de générations = 3000 Nombre d'exécutions = 30					

5.3.1.2 Les problèmes de référence

Pour évaluer l'efficacité de IMOGWO en tant qu'optimiseur multi-objectifs, il a été testé sur 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. De plus, ils offrent différents espaces de recherche avec des formes et des caractéristiques variées de PF_{true} : linéaires, multimodaux, concaves, biaisés, mixtes, déconnectés et "scaled". La résolution des problèmes présentant différentes caractéristiques (DTLZ1-4,7) est évaluée dans cette étude. Ils sont résolus avec un nombre d'objectifs égal à 2, 3, 4, 6 et 8. Le tableau 5.5 décrit les caractéristiques des problèmes examinés. Les formules mathématiques appliquées dans les problèmes DTLZ1-4,7 et leurs paramètres sont représentés dans le tableau 5.6.

TABLEAU 5.5 – Caractéristiques des problèmes DTLZ1-4,7 étudiés

Problème	**Caractéristiques**
DTLZ1	Linéaire, multimodal
DTLZ2	Concave
DTLZ3	Concave, multimodal
DTLZ4	Concave, biaisé
DTLZ7	Mixte, déconnecté, multimodal, "scaled"

TABLEAU 5.6 – Formules mathématiques et paramètres des problèmes DTLZ1-4,7 étudiés

Problème	Formules mathématiques	Paramètres	Domaines	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	mêmes $f_{1:M}$ que DTLZ2 $g = 100[k + \sum_{i=1}^{k}((x_i-0.5)^2 - \cos(20\pi(x_i-0.5)))]$	$k = 10$		
DTLZ4	mêmes équations que DTLZ2, sauf que tous les x_i sont remplacés par 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 Les indicateurs de performance

Les indicateurs IGD et NHV (section 3.2.5) ont été sélectionnés pour leur large utilisation en tant que critères d'évaluation, fournissant une vision complète des performances de convergence et de

distribution de PF [76]. Ces critères sont donc appropriés pour évaluer un algorithme d'optimisation.

5.3.2 Résultats et discussion

Dans cette section, nous présentons les résultats de l'évaluation numérique des algorithmes étudiés appliqués à la résolution des problèmes DTLZ1-4,7. Les valeurs de IGD et NHV obtenues sont représentées graphiquement et comparées.
Les tableaux 5.7, 5.8, 5.9, 5.10 et 5.11 montrent les valeurs moyennes, médianes, meilleures (minimum) et moins bonnes (maximum) fournies par les indicateurs IGD et NHV après la résolution des problèmes DTLZ1-4,7 pour chaque nombre d'objectifs. Ils incluent également l'écart-type des différentes valeurs IGD et NHV obtenues lors des 30 tests réalisés.
Ces valeurs représentent les résultats expérimentaux obtenus en appliquant IMOGWO et les autres algorithmes comparatifs. Nous utilisons les valeurs moyennes des deux indicateurs pour comparer les performances des optimiseurs dans cette section. Les meilleures valeurs moyennes de IGD et NHV entre celles données par IMOGWO et MOGWO sont soulignées. Cependant, les meilleures valeurs moyennes parmi celles obtenues par tous les optimiseurs pour chaque problème et chaque nombre d'objectifs sont mises en gras.
Les résultats expérimentaux confirment que IMOGWO a surpassé MOGWO dans tous les cas de test considérés (problèmes DTLZ1-4,7) et pour toutes les configurations d'objectifs étudiées (2, 3, 4, 6, 8). Les valeurs de IGD et NHV obtenues avec IMOGWO sont inférieures à celles obtenues avec MOGWO, indiquant ainsi une meilleure convergence et distribution des fronts de Pareto avec IMOGWO.
Cette amélioration est clairement visible à travers les courbes présentant la variation de IGD et NHV pour chaque problème. Ces courbes, illustrées dans les figures 5.6 et 5.7, montrent les valeurs moyennes de IGD et NHV en fonction du nombre d'objectifs. Dans chaque problème DTLZ1-4,7, la figure 5.6 comprend deux graphiques (a) et (b). Le graphique (a) affiche la variation des valeurs de IGD pour tous les algorithmes étudiés. Les résultats d'ESPEA sont plus élevés que ceux des autres optimiseurs, rendant la visualisation du graphique difficile. Ainsi, le graphique (b) exclut les valeurs fournies par ESPEA. De plus, les tableaux 5.7, 5.8, 5.9, 5.10 et 5.11 indiquent que IMOGWO présente généralement les meilleures valeurs moyennes de IGD et NHV.
Pour le problème DTLZ1, IMOGWO affiche de meilleurs résultats IGD pour toutes les configurations du nombre d'objectifs examinées (tableau 5.7 et figure 5.6). De plus, il présente un NHV inférieur à tous les autres optimiseurs pour un nombre d'objectifs égal à 2, 3, 4 et 6 (tableau 5.7 et figure 5.7). En revanche, pour un nombre d'objectifs égal à 8, c'est MOEA/DD qui obtient la meilleure valeur moyenne de NHV. La capacité d'IMOGWO à offrir une convergence et une dis-

TABLEAU 5.7 – Valeurs de IGD et NHV obtenues sur DTLZ1

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

tribution efficaces est mise en évidence dans la figure 5.8. Cette représentation montre les $PF\text{s}_{true}$ ainsi que les solutions obtenues (les PFs) en utilisant IMOGWO sur DTLZ1 pour des nombres d'objectifs allant de 2 à 8. Elle indique également que les PFs obtenus restent proches des $PF\text{s}_{true}$ de DTLZ1 malgré une dégradation des performances due à l'augmentation du nombre d'objectifs. Pour le problème DTLZ2, les résultats d'IGD et de NHV obtenus (voir tableau 5.8, figure 5.6 et figure 5.7) démontrent clairement la supériorité de IMOGWO par rapport à MOGWO pour tous les nombres d'objectifs considérés : 2, 3, 4, 6 et 8. Initialement, en termes d'IGD, IMOGWO surclasse tous les autres algorithmes comparatifs pour 3 et 4 objectifs. Cependant, les meilleures valeurs moyennes d'IGD ont été fournies par ESPEA pour 2 objectifs, et par MOEA/D-IEps pour 6 et 8 objectifs. Malgré cela, IMOGWO a présenté la plus petite meilleure valeur d'IGD pour 2 objectifs. Par contre, MOGWO, DMOPSO et MOEA/DD ont donné les pires résultats pour tous les nombres d'objectifs. Ensuite, en ce qui concerne le NHV, IMOGWO s'avère être le meilleur pour 4 objectifs. Cependant, pour 2 et 3 objectifs, ESPEA a fourni les plus petites valeurs moyennes de

TABLEAU 5.8 – Valeurs de IGD et NHV obtenues sur DTLZ2

Algorithme	Valeur	2 objectifs		3 objectifs		4 objectifs		6 objectifs		8 objectifs	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Moyenne	0.000196	0.021344	**0.000604**	0.159711	**0.010474**	**0.166278**	0.021944	0.315858	0.028232	0.360284
	Médiane	0.000197	0.022155	0.000605	0.161147	0.010319	0.164841	0.022028	0.317608	0.028215	0.358880
	Écart-type	0.000031	0.002890	0.000029	0.007295	0.000574	0.016177	0.000513	0.010170	0.000273	0.006830
	Pire	0.000262	0.025427	0.000648	0.175340	0.012347	0.206035	0.022724	0.332887	0.028613	0.371265
	Meilleur	0.000149	0.016349	0.000538	0.144363	0.009559	0.132658	0.020194	0.275735	0.027443	0.344187
MOGWO	Moyenne	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023146	0.343687	0.028614	0.378260
	Médiane	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023144	0.343757	0.028613	0.378066
	Écart-type	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000006	0.000516	0.000002	0.000428
	Pire	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023165	0.344740	0.028623	0.380165
	Meilleur	0.000304	0.029766	0.000751	0.205355	0.012615	0.217063	0.023138	0.342908	0.028613	0.378066
DMOPSO	Moyenne	0.000173	0.018370	0.000816	0.208090	0.016303	0.275250	0.026875	0.281056	0.030979	0.357137
	Médiane	0.000173	0.018364	0.000815	0.207933	0.016304	0.275839	0.026824	0.279158	0.031002	0.346567
	Écart-type	0.000000	0.000032	0.000005	0.003880	0.000095	0.001965	0.000372	0.020820	0.001784	0.023997
	Pire	0.000173	0.018435	0.000825	0.215057	0.016482	0.278648	0.027579	0.334046	0.033999	0.398225
	Meilleur	0.000173	0.018318	0.000805	0.200396	0.016081	0.270695	0.026235	0.245394	0.026439	0.325745
MOEA/D-IEp	Moyenne	0.000162	0.018995	0.000790	0.218628	0.011813	0.184438	**0.021052**	0.166039	**0.023672**	0.225824
	Médiane	0.000162	0.018904	0.000791	0.216210	0.011773	0.187251	0.021078	0.165951	0.023498	0.228383
	Écart-type	0.000003	0.000437	0.000033	0.016475	0.000498	0.020268	0.000536	0.020099	0.000731	0.031303
	Pire	0.000167	0.019969	0.000890	0.266657	0.013204	0.216263	0.022190	0.204615	0.026211	0.278026
	Meilleur	0.000157	0.018338	0.000734	0.183085	0.010797	0.138909	0.019962	0.128972	0.022754	0.159115
MOEA/DD	Moyenne	0.000173	0.018250	0.000667	0.141488	0.019390	0.217154	0.021999	**0.081595**	0.026414	**0.000000**
	Médiane	0.000173	0.018250	0.000667	0.141493	0.019390	0.217153	0.022012	0.081681	0.026414	0.000000
	Écart-type	0.000000	0.000000	0.000000	0.000026	0.000000	0.000002	0.000078	0.000424	0.000000	0.000000
	Pire	0.000173	0.018250	0.000667	0.141513	0.019391	0.217159	0.022102	0.082045	0.026414	0.000000
	Meilleur	0.000173	0.018250	0.000667	0.141352	0.019389	0.217149	0.021591	0.079428	0.026413	0.000000
ESPEA	Moyenne	**0.000157**	**0.017179**	0.000658	**0.129807**	0.015704	0.283479	0.162279	1.000000	0.140856	1.000000
	Médiane	0.000156	0.017125	0.000657	0.130333	0.015743	0.272516	0.162262	1.000000	0.140841	1.000000
	Écart-type	0.000002	0.000267	0.000011	0.004172	0.000936	0.059490	0.000328	0.000000	0.000190	0.000000
	Pire	0.000163	0.017710	0.000681	0.140629	0.017810	0.470555	0.162788	1.000000	0.141209	1.000000
	Meilleur	0.000154	0.016779	0.000638	0.121541	0.014164	0.194287	0.161610	1.000000	0.140434	1.000000

NHV, tandis que IMOGWO a donné la plus petite meilleure valeur de NHV pour 2 objectifs. Pour 6 et 8 objectifs, les résultats de NHV montrent que MOEA/DD est le plus efficace. En revanche, les valeurs NHV les plus élevées (les pires) ont été obtenues en appliquant MOGWO, DMOPSO et MOEA/D-IEps pour tous les nombres d'objectifs. Bien que IMOGWO ne soit pas le meilleur dans tous les cas de test de DTLZ2, les solutions obtenues sont proches des valeurs réelles des fronts de Pareto et sont bien distribuées, comme le montre la Figure 5.9.

Les résultats obtenus avec DTLZ3 (tableau 5.9, figure 5.6 et figure 5.7) illustrent une amélioration des indicateurs IGD et NHV grâce à l'utilisation de IMOGWO par rapport à MOGWO dans toutes les configurations de test. Ces résultats fournissent également un aperçu des performances de tous les optimiseurs évalués. IMOGWO se distingue particulièrement pour un nombre d'objectifs égal à 2, 3 et 8, avec des valeurs d'IGD les plus basses parmi tous les algorithmes testés. Pour 4 et 6 objectifs, les algorithmes les plus performants sont respectivement MOEA/D-IEps et MOEA/DD. En termes de NHV, IMOGWO offre les meilleures performances pour 2 objectifs,

TABLEAU 5.9 – Valeurs de IGD et NHV obtenues sur DTLZ3

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

tandis qu'ESPEA domine pour 3 et 4 objectifs. Pour les cas de 6 et 8 objectifs, c'est MOEA/DD qui affiche la meilleure efficacité. La distribution des fronts de Pareto obtenue par IMOGWO après la résolution de DTLZ3 est présentée dans la figure 5.10, soulignant la capacité de IMOGWO à converger efficacement vers PF_{true} et à maintenir une distribution appropriée des solutions, en particulier pour 2, 3, 4 et 6 objectifs.

Le quatrième problème, DTLZ4, met en évidence les performances d'IMOGWO. Les PF_{true} pour des nombres d'objectifs de 2, 3, 6 et 8 sont illustrés dans la figure 5.11. Leur comparaison avec les résultats obtenus par IMOGWO montre une convergence et une distribution satisfaisantes dans la plupart des cas. Les valeurs d'IGD et de NHV ainsi que leur évolution sont présentées dans le tableau 5.10, la figure 5.6 et la figure 5.7. IMOGWO surpasse MOGWO en termes d'IGD et de NHV dans tous les scénarios testés. Notamment, IMOGWO se distingue comme l'optimiseur le plus efficace pour le plus grand nombre d'objectifs considéré, 8. Pour DTLZ4 avec 4 objectifs, IMOGWO affiche la meilleure performance en termes d'IGD. Cependant, ses performances sont

TABLEAU 5.10 – Valeurs de IGD et NHV obtenues sur DTLZ4

Algorithme	Valeur	2 objectifs		3 objectifs		4 objectifs		6 objectifs		8 objectifs	
		IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV	IGD	NHV
IMOGWO	Moyenne	0.000506	0.054129	0.001684	0.360809	0.019157	0.319157	0.037798	0.633899	**0.032461**	0.635780
	Médiane	0.000451	0.051394	0.001711	0.371769	0.019393	0.284616	0.038530	0.656723	0.032051	0.661102
	Écart-type	0.000178	0.014713	0.000322	0.099573	0.001315	0.087121	0.002542	0.124194	0.001670	0.116725
	Pire	0.001197	0.095778	0.002323	0.492429	0.021256	0.533864	0.040638	0.813541	0.036879	0.772086
	Meilleur	0.000356	0.034310	0.000940	0.142299	0.016527	0.213991	0.029180	0.390138	0.030236	0.282356
MOGWO	Moyenne	0.004026	0.341989	0.002735	0.504163	0.042100	0.643649	0.053683	0.814735	0.046146	0.772086
	Médiane	0.004026	0.341989	0.002735	0.504163	0.042099	0.643604	0.053671	0.813936	0.046149	0.772086
	Écart-type	0.000000	0.000000	0.000000	0.000000	0.000001	0.000204	0.000180	0.001360	0.000006	0.000000
	Pire	0.004026	0.341989	0.002735	0.504163	0.042102	0.644594	0.054053	0.817091	0.046154	0.772086
	Meilleur	0.004026	0.341989	0.002735	0.504163	0.042099	0.643521	0.053482	0.813626	0.046137	0.772086
DMOPSO	Moyenne	0.000161	0.019825	0.001002	0.114134	0.024345	0.311998	0.040914	0.453365	0.040932	0.493325
	Médiane	0.000160	0.019792	0.000994	0.112846	0.024118	0.305437	0.040922	0.450765	0.041036	0.495337
	Écart-type	0.000002	0.000236	0.000066	0.009308	0.001305	0.021361	0.001952	0.037820	0.001369	0.043626
	Pire	0.000166	0.020302	0.001157	0.139359	0.027239	0.366492	0.045388	0.537338	0.044053	0.587461
	Meilleur	0.000159	0.019321	0.000858	0.098928	0.022119	0.286135	0.037274	0.374113	0.038430	0.400545
MOEA/D-IEp	Moyenne	0.000163	0.019675	0.001457	0.138597	0.021770	**0.155166**	0.033358	0.131438	0.037714	0.136173
	Médiane	0.000163	0.019610	0.001481	0.139313	0.021789	0.157871	0.034416	0.118508	0.037615	0.132807
	Écart-type	0.000002	0.000392	0.000117	0.018923	0.001745	0.014225	0.003220	0.038455	0.001012	0.021995
	Pire	0.000167	0.020643	0.001690	0.195115	0.024814	0.177411	0.038190	0.196062	0.039941	0.180964
	Meilleur	0.000160	0.018942	0.001280	0.103931	0.018104	0.126861	0.026377	0.074517	0.035624	0.102402
MOEA/DD	Moyenne	0.000156	0.018604	0.000988	0.060576	**0.019090**	0.217353	**0.033271**	**0.123987**	0.035214	**0.000000**
	Médiane	0.000156	0.018604	0.000988	0.060579	0.019090	0.217353	0.035656	0.129790	0.036414	0.000000
	Écart-type	0.000000	0.000000	0.000000	0.000010	0.000000	0.000002	0.004304	0.014266	0.002400	0.000000
	Pire	0.000156	0.018604	0.000988	0.060597	0.019092	0.217357	0.039953	0.129792	0.037414	0.000000
	Meilleur	0.000156	0.018604	0.000988	0.060555	0.019089	0.217349	0.025946	0.076907	0.031414	0.000000
ESPEA	Moyenne	**0.000150**	**0.017512**	**0.000609**	**0.051805**	0.019296	0.226526	0.162201	1.000000	0.141011	1.000000
	Médiane	0.000150	0.017502	0.000611	0.052796	0.019342	0.216586	0.162373	1.000000	0.141010	1.000000
	Écart-type	0.000001	0.000265	0.000012	0.005050	0.000459	0.033544	0.000538	0.000000	0.000183	0.000000
	Pire	0.000152	0.018299	0.000626	0.064094	0.019999	0.297932	0.162887	1.000000	0.141368	1.000000
	Meilleur	0.000148	0.017008	0.000581	0.042291	0.018197	0.176084	0.160633	1.000000	0.140635	1.000000

plus faibles pour les autres nombres d'objectifs. ESPEA montre les plus faibles valeurs d'IGD et de NHV pour 2 et 3 objectifs. Pour 4 objectifs, les meilleures valeurs d'IGD et de NHV sont fournies respectivement par MOEA/DD et MOEA/D-IEps. Avec 6 objectifs, MOEA/DD offre les meilleures valeurs pour les indicateurs évalués. De même, MOEA/DD présente la plus petite valeur de NHV pour 8 objectifs.

Le dernier problème examiné dans nos expériences est DTLZ7. Avec ses PF_{true} mixtes, déconnectés, multimodaux et "scaled", il représente un défi supplémentaire en termes de complexité. Les résultats d'IGD et de NHV présentés dans le tableau 5.11 et illustrés dans les figures 5.6 et 5.7 confirment la supériorité d'IMOGWO dans la résolution de DTLZ7 par rapport aux autres optimiseurs. Premièrement, IMOGWO surpasse nettement MOGWO en termes d'indicateurs IGD et NHV pour tous les cas de test. Deuxièmement, comparé aux autres méthodes, IMOGWO fournit les valeurs d'IGD les plus faibles (meilleures) pour 2, 4, 6 et 8 objectifs. Pour 3 objectifs, ESPEA obtient la meilleure moyenne d'IGD. Par ailleurs, IMOGWO présente la plus petite meilleure valeur

TABLEAU 5.11 – Valeurs de IGD et NHV obtenues sur DTLZ7

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

d'IGD pour 3 objectifs. Les résultats NHV confirment également la performance de IMOGWO, qui fournit les valeurs les plus faibles pour tous les nombres d'objectifs considérés. Les défis posés par les PF_{true} de DTLZ7 sont illustrés dans la figure 5.12. Les résultats montrent que malgré la complexité du problème, IMOGWO parvient à maintenir une convergence élevée, bien que sa distribution diminue légèrement pour 2 et 3 objectifs.

Pour mieux illustrer les résultats obtenus, la Figure 5.13 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

FIGURE 5.6 – Valeurs IGD moyennes pour différents nombres d'objectifs et différents problèmes de référence

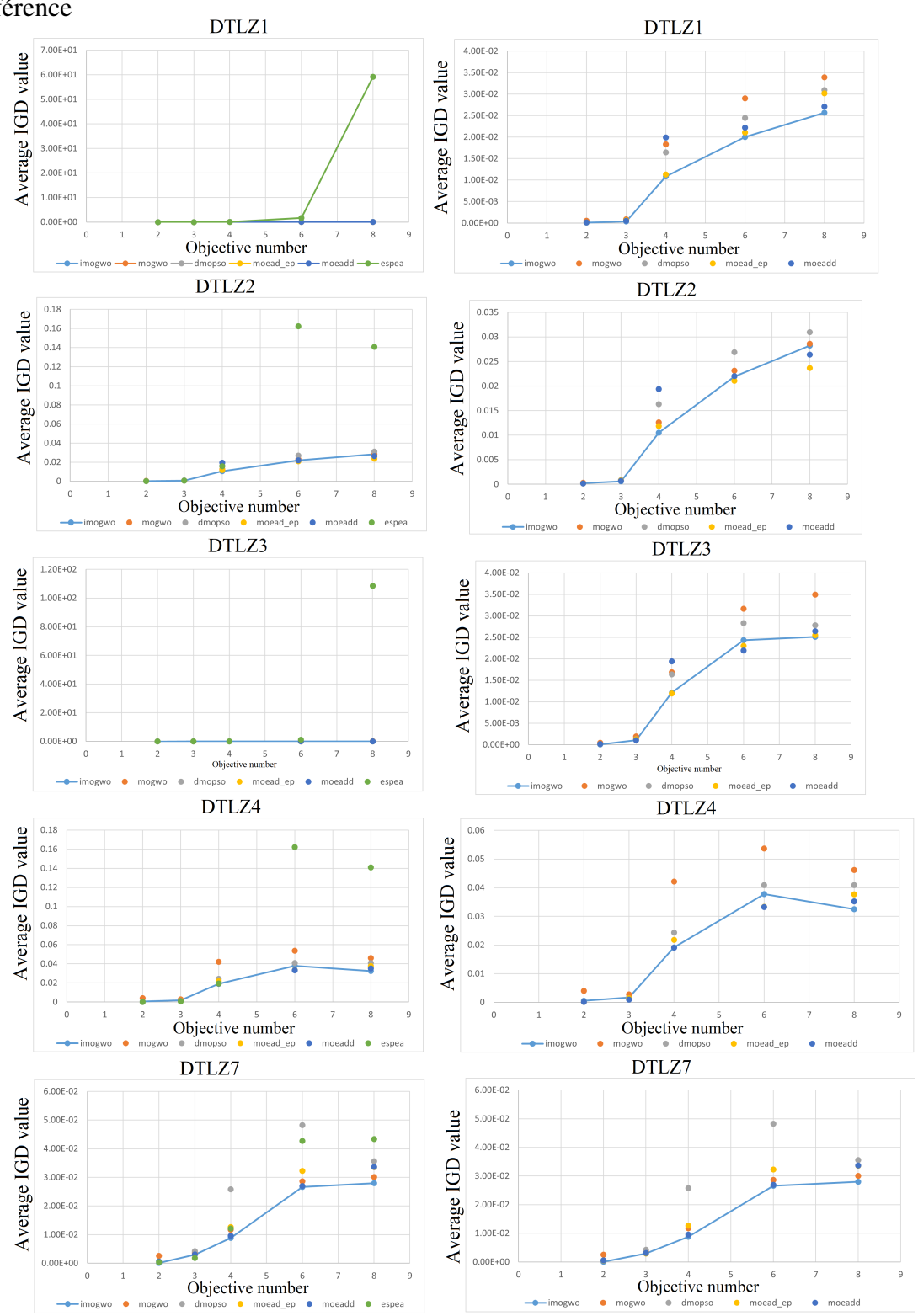
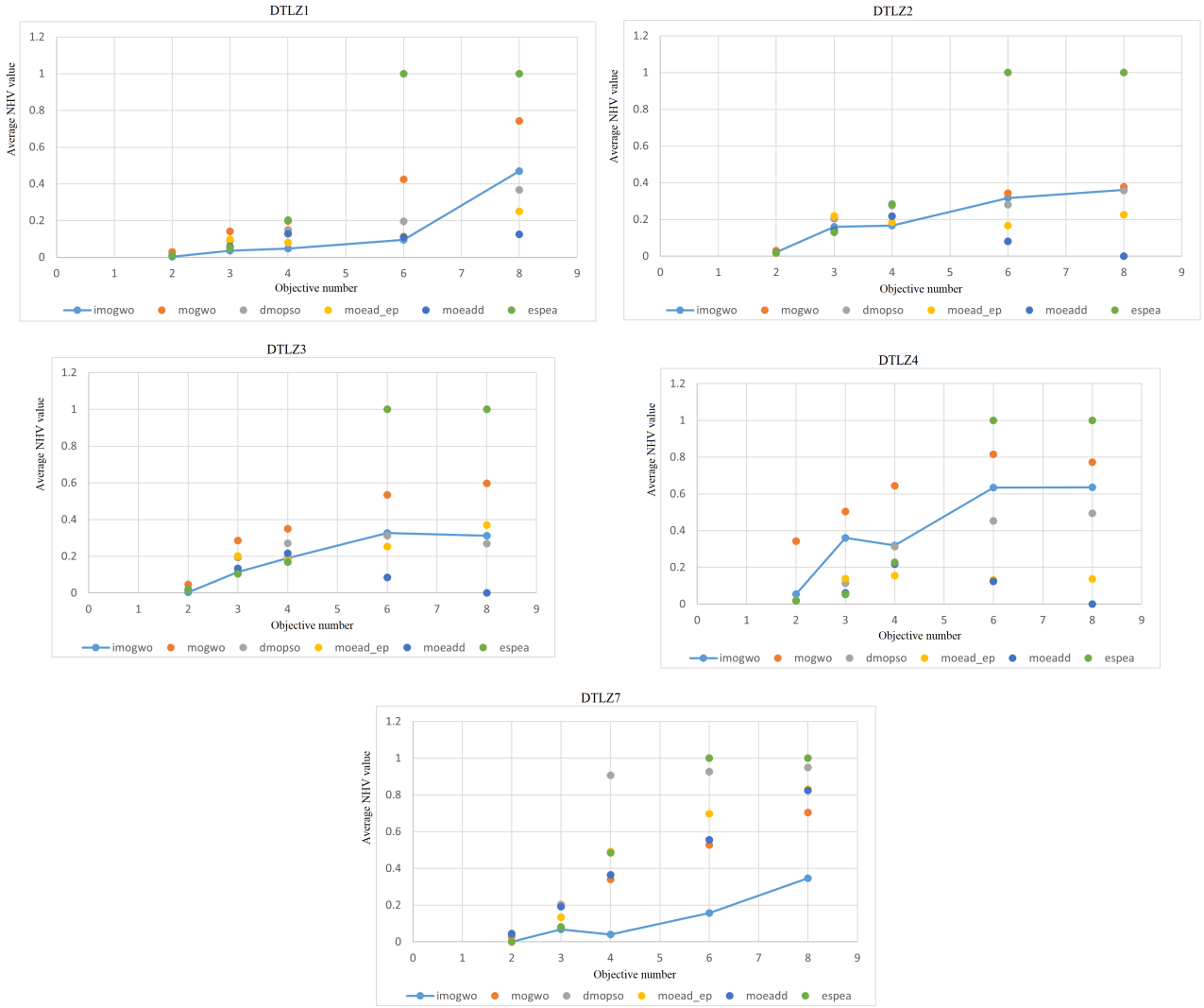


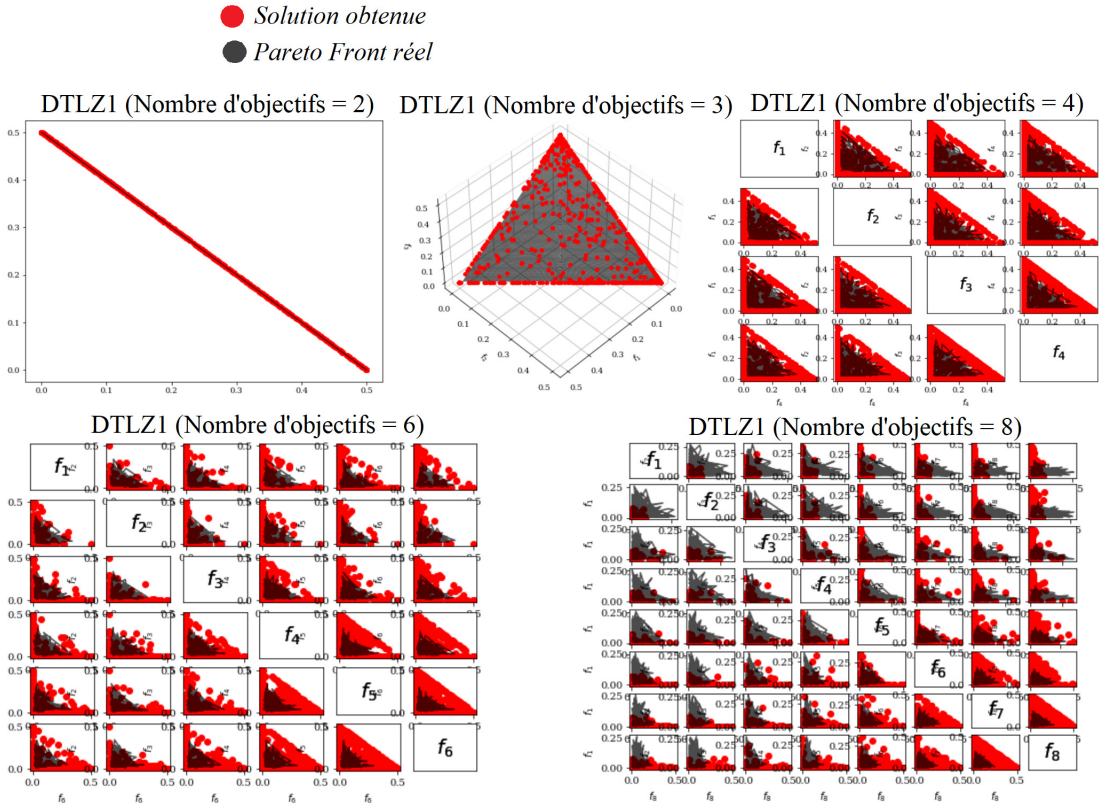
FIGURE 5.7 – Valeurs NHV moyennes pour différents nombres d'objectifs et différents problèmes de référence



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.

La comparaison des performances des optimiseurs est également présentée dans la figure 5.14. Cette dernière expose les résultats expérimentaux obtenus à travers les indicateurs IGD et NHV pour l'ensemble des problèmes DTLZ1-4,7, ainsi que pour les différentes configurations d'objectifs étudiées. Elle se décompose en 3 graphiques distincts. Le premier graphique (a) de la figure 5.14 illustre les résultats globaux de l'IGD obtenus en appliquant tous les optimiseurs dans chaque scénario de test. Le graphique (b) montre ces résultats en excluant ESPEA, dont les valeurs d'IGD élevées rendent la visualisation moins claire. Enfin, le graphique (c) met en évidence les résultats globaux de l'NHV pour tous les algorithmes étudiés dans chaque configuration de test. Les

FIGURE 5.8 – Les "Pareto fronts" réels et les solutions (PFs) fournies par IMOGWO pour DTLZ1



diagrammes en boîte à moustaches utilisés mettent en évidence l'amélioration significative apportée par IMOGWO par rapport à MOGWO, tant en termes d'IGD que de NHV, renforçant ainsi la convergence et la distribution des solutions. Comparé aux autres optimiseurs, IMOGWO est le meilleur en termes des valeurs moyennes et médianes de l'IGD. De même, ses valeurs minimales, médianes et moyennes de l'indicateur NHV sont parmi les meilleures malgré la concurrence d'autres optimiseurs.

Par conséquent, à partir de ces représentations graphiques, il est possible de conclure que, bien que l'algorithme IMOGWO ne soit pas toujours le meilleur en termes de performances sur les problèmes de référence étudiés, il est le meilleur dans la majorité des tests réalisés, notamment pour DTLZ1 et DTLZ7. Tableau 5.12 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.

Ainsi, pour vérifier si IMOGWO a présenté une amélioration notable par rapport aux autres optimiseurs, des tests statistiques inférentiels, décrits dans la sous-section suivante, ont été appliqués.

FIGURE 5.9 – Les "Pareto fronts" réels et les solutions (PFs) fournies par IMOGWO pour DTLZ2

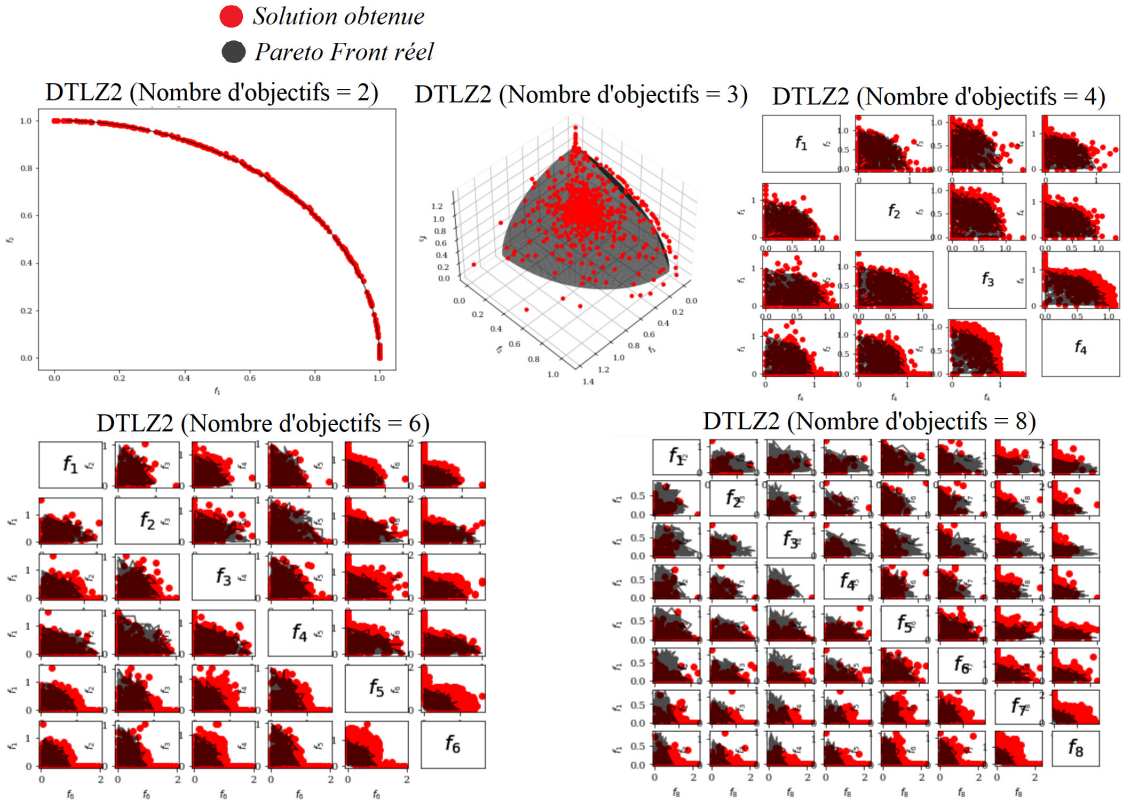


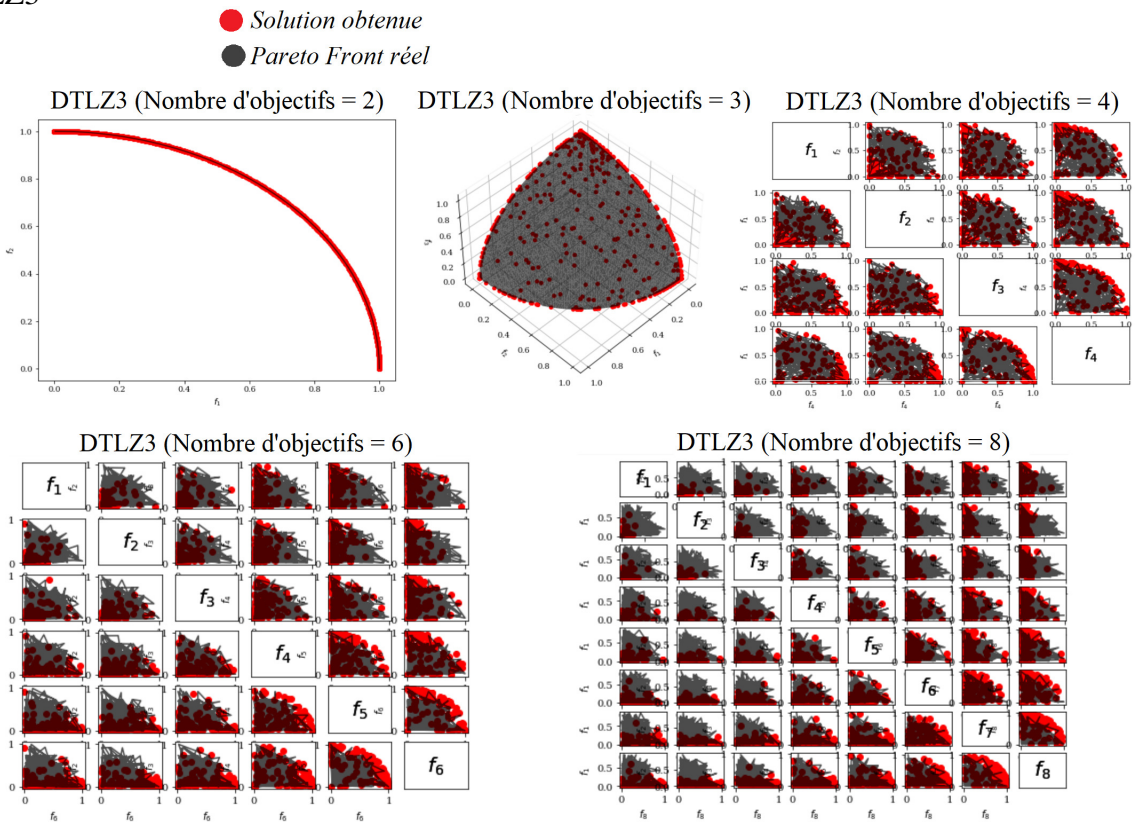
TABLEAU 5.12 – Le nombre de fois où chaque algorithme a fourni les meilleures valeurs moyennes des indicateurs IGD et NHV

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

5.3.3 Tests statistiques inférentiels

Pour évaluer les performances des divers algorithmes, des méthodes de tests statistiques ont été utilisées sur les résultats des indicateurs IGD et NHV. En général, les tests statistiques sont employés pour déterminer si un nouvel algorithme proposé surpasse les algorithmes existants dans la résolution d'un problème donné. Dans ce cadre, plusieurs procédures de comparaison statistique ont été proposées. [149] a présenté en détail de nombreux tests possibles et a fourni un tutoriel utile sur l'application de ces méthodes en intelligence computationnelle. Ainsi, puisque l'objectif de ce travail est de comparer l'algorithme proposé avec ceux existants, des tests basés sur des comparaisons (1 x N) ont été réalisés. Le test de Friedman avec extension Iman-Davenport et la procédure post-hoc de Holm ont été choisis comme tests de comparaison multiple pour les expériences réalisées, car ils sont recommandés par [149]. Les prochaines sections détaillent les étapes suivies pour appliquer ces deux méthodes.

FIGURE 5.10 – Les "Pareto fronts" réels et les solutions (PFs) fournies par IMOGWO pour DTLZ3



5.3.3.1 Application du test de Friedman avec extension Iman-Davenport

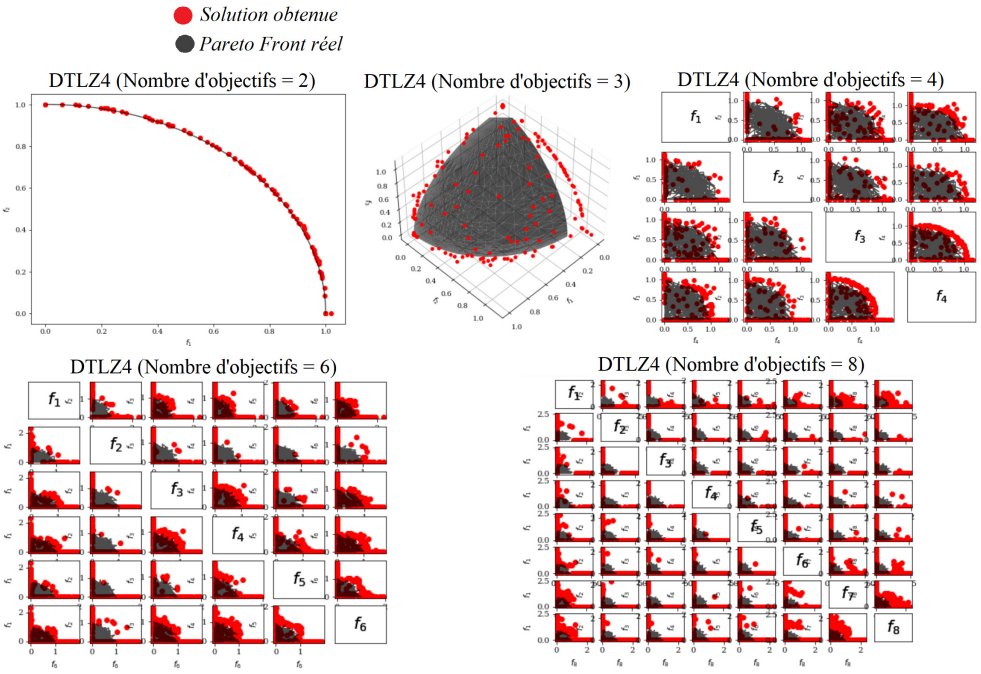
Le test de Friedman avec l'extension d'Iman-Davenport est utilisé pour comparer un algorithme à un ensemble d'autres algorithmes. Son objectif est de déterminer s'il existe une différence statistiquement significative entre les méthodes évaluées. Pour comparer IMOGWO à d'autres optimiseurs, les étapes suivantes sont suivies :

1. Les optimiseurs sont classés pour chaque problème de test en fonction de leurs performances en termes d'IGD et de NHV, où le meilleur résultat obtient le rang le plus bas.
2. La moyenne des rangs de chaque optimiseur sur l'ensemble des problèmes de test est calculée pour obtenir le rang final R_j (Eq. 5.1) :

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

n représente le nombre de cas de test étudiés, qui est de 25. r_i est le rang de l'optimiseur sur le cas de test i.

FIGURE 5.11 – Les "Pareto fronts" réels et les solutions (PFs) fournies par IMOGWO pour DTLZ4



3. Les statistiques de Friedman sont calculées en utilisant l'équation 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 représente le nombre d'algorithmes comparés, qui est de 6.

4. Les statistiques d'Iman-Davenport sont calculées en utilisant l'équation 5.3 :

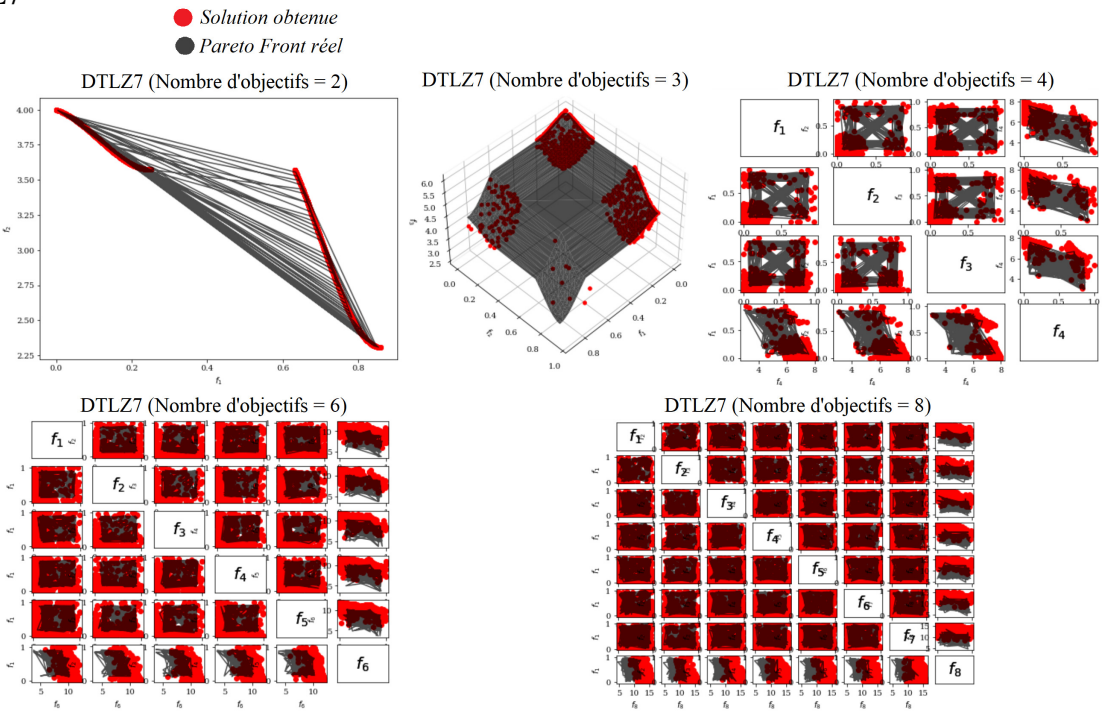
$$F_{id} = \frac{(n-1)F_f^2}{n(k-1) - F_f^2} \tag{5.3}$$

5. La valeur correspondante dans la table de distribution F est obtenue. Dans notre étude, cette valeur est égale à $F(5, 120) = 2.29$.

6. $F(5, 120)$ est comparé aux statistiques obtenues. Si $F(5, 120)$ est inférieur à F_f et F_{id}, l'hypothèse nulle de Friedman (tous les algorithmes ont la même performance) est rejetée. Ainsi, on peut conclure qu'il existe une différence significative entre les algorithmes comparés en termes de valeurs de l'indicateur de performance.

Le tableau 5.19 présente les rangs moyens obtenus par les optimiseurs sur l'ensemble des problèmes de test pour les métriques IGD et NHV. Le rang le plus bas (le meilleur) est indiqué en gras, tandis que le rang suivant est souligné. Ce tableau inclut également les statistiques de Fried-

FIGURE 5.12 – Les "Pareto fronts" réels et les solutions (PFs) fournies par IMOGWO pour DTLZ7



man et d'Iman-Davenport, ainsi que les valeurs correspondantes dans la table de distribution F pour les métriques IGD et NHV. En ce qui concerne l'indicateur IGD, IMOGWO obtient le meilleur rang moyen parmi les algorithmes comparés. La valeur critique de la distribution F (5,120), égale à 2.29, est inférieure aux valeurs des statistiques de Friedman et d'Iman-Davenport (respectivement 42.22 et 12.24). Par conséquent, l'hypothèse nulle de Friedman est rejetée, ce qui indique que les algorithmes comparés présentent des résultats IGD significativement différents.
En ce qui concerne l'indicateur NHV, MOEA/DD obtient le meilleur rang moyen, suivi de près par IMOGWO. De plus, les statistiques de Friedman et d'Iman-Davenport sont respectivement de 30.51 et 7.75. Ces valeurs statistiques dépassent la valeur critique de la distribution F (2.29), ce qui souligne une différence substantielle entre les algorithmes comparés en termes de résultats NHV. Comme les résultats du test de Friedman avec extension d'Iman-Davenport indiquent une différence significative entre les optimiseurs étudiés en termes d'indicateurs IGD et NHV, nous procéderons à l'application du test post-hoc de Holm [149].

5.3.3.2 Application de la procédure post-hoc de Holm

La procédure post-hoc de Holm peut être utilisée pour comparer un algorithme à deux ou plusieurs autres algorithmes. Pour comparer IMOGWO à d'autres optimiseurs, cette procédure consiste en :

FIGURE 5.13 – Valeurs moyennes des indicateurs IGD et NHV sur tous les problèmes DTLZ1-4,7 et les différents nombres d'objectifs étudiés

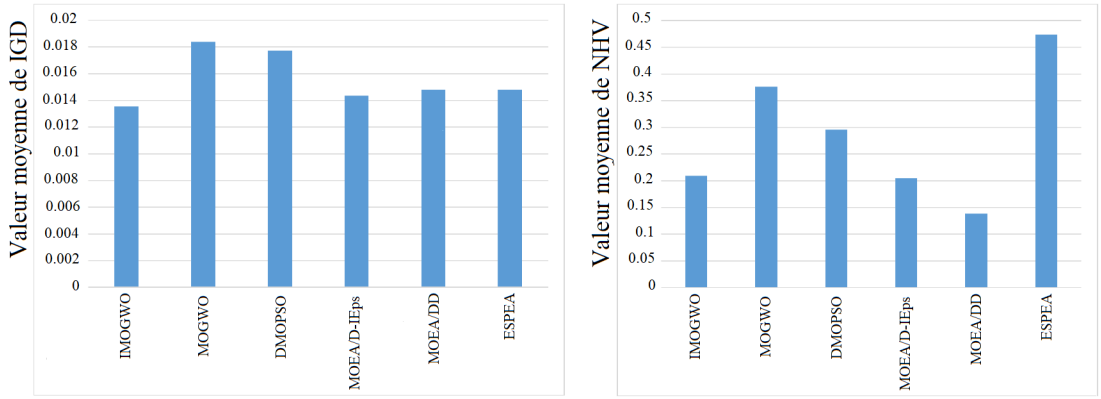
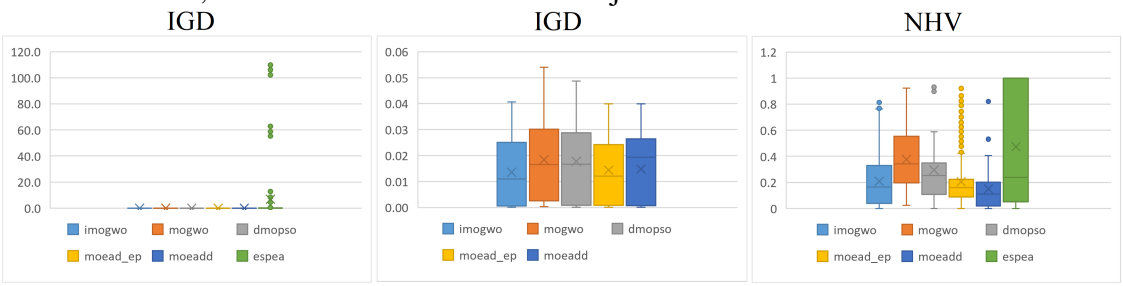


FIGURE 5.14 – Performance des optimiseurs testés en termes d'indicateurs IGD et NHV pour les problèmes DTLZ1-4,7 et les différents nombres d'objectifs étudiés



(a) (b) (c)

1. Calculer les valeurs z pour les optimiseurs étudiés en utilisant l'équation 5.4 :

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

   Un résultat négatif de z signifie que IMOGWO surpasse l'autre optimiseur.

2. Classer les optimiseurs comparés selon les valeurs z (du plus grand au plus petit). r_i représente le rang de l'optimiseur i en fonction du z_i correspondant.

3. Calculer les valeurs de Holm pour les optimiseurs comparés en utilisant l'équation 5.5 :

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

4. Calculer les valeurs p qui correspondent aux valeurs z. Les valeurs p sont basées sur la distribution normale et les valeurs de Holm. Si la valeur p d'un optimiseur est inférieure à la valeur de Holm correspondante, alors l'hypothèse nulle sera rejetée. En d'autres termes, il y a une différence significative entre cet optimiseur et IMOGWO en termes de valeurs de l'indicateur de performance.

TABLEAU 5.13 – Les rangs moyens des optimiseurs et le calcul des statistiques de Friedman et d'Iman-Davenport pour les métriques IGD et NHV

Algorithme		IMOGWO	MOGWO	DMOPSO	MOEA/D-IEps	MOEA/DD	ESPEA
Rang moyen	IGD	**1.92**	4.84	4.44	3.02	2.9	3.88
	NHV	2.64	4.88	3.96	3.16	**2.44**	3.92
Statistique de Friedman pour IGD = 42.22 Statistique de Iman-Davenport pour IGD = 12.24 F(5,120) = 2.29							
Statistique de Friedman pour NHV = 30.51 Statistique de Iman-Davenport pour NHV = 7.75 F(5,120) = 2.29							

Les résultats des tests de la procédure post-hoc de Holm, en termes d'indicateurs IGD et NHV, sont présentés dans les tableaux 5.14 et 5.15 respectivement. Nous notons tout d'abord que pour l'indicateur IGD (tableau 5.14), toutes les valeurs de z trouvées sont négatives, indiquant que le test de Holm confirme que IMOGWO surpasse tous les autres optimiseurs. De plus, les valeurs p de MOGWO, DMOPSO et ESPEA sont inférieures aux valeurs de Holm correspondantes. Par conséquent, nous pouvons conclure que les résultats IGD de IMOGWO sont significativement meilleurs que ceux de MOGWO, DMOPSO et ESPEA. Dans le même contexte, bien que les valeurs de z calculées pour MOEA/D-IEps et MOEA/DD soient négatives, leurs valeurs p sont supérieures à leurs valeurs de Holm correspondantes. Autrement dit, IMOGWO surpasse légèrement MOEA/D-IEps et MOEA/DD en termes d'indicateur IGD.

En ce qui concerne l'indicateur NHV, les résultats du test de Holm révèlent que les valeurs calculées de z sont majoritairement négatives, à l'exception de celle de MOEA/DD. Autrement dit, concernant le NHV (Tableau 5.15), IMOGWO surpasse les autres optimiseurs, à l'exception de MOEA/DD. De plus, les valeurs p de MOGWO et d'ESPEA sont inférieures aux valeurs de Holm, ce qui indique que les performances d'IMOGWO sont nettement meilleures que celles de MOGWO et d'ESPEA. Cependant, les valeurs p de DMOPSO et de MOEA/D-IEps dépassent les valeurs de Holm, montrant que les performances d'IMOGWO sont légèrement supérieures à celles de DMOPSO et de MOEA/D-IEps en termes de l'indicateur NHV. En résumé, la différence entre leurs résultats NHV n'est pas très marquée.

TABLEAU 5.14 – Résultats du test post-hoc de Holm pour la comparaison entre IMOGWO et les autres optimiseurs en termes de l'indicateur IGD

Optimiseur	$\mathbf{z_i}$	$\mathbf{r_i}$	**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

TABLEAU 5.15 – Résultats du test post-hoc de Holm pour la comparaison entre IMOGWO et les autres optimiseurs en termes de l'indicateur NHV

Optimiseur	$\mathbf{z_i}$	$\mathbf{r_i}$	**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 Synthèse

Dans cette section, nous utilisons les métriques IGD et NHV pour évaluer les performances de IMOGWO par rapport à un ensemble d'optimiseurs récemment développés et largement utilisés. Les résultats expérimentaux 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 fronts de Pareto. 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 DLH a renforcé l'exploration et la distribution des solutions obtenues par IMOGWO.

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 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.

5.4 Résultats expérimentaux et discussion de la localisation en intérieur en hybridant le DL et les métaheuristiques

Une série de tests a été réalisée afin d'évaluer l'efficacité des contributions décrites dans les sections 4.5 et 4.6 du chapitre 4 : (i) l'optimisation des poids d'un modèle DL par un algorithme d'optimisation métaheuristique et (ii) l'application de cette approche pour développer un modèle DL de localisation en intérieur basé sur les mesures UWB ToF. Cette section commence par décrire la configuration expérimentale, puis elle analyse les résultats obtenus.

5.4.1 Configuration Expérimentale

Cette section détaille les expérimentations menées, notamment le dataset utilisé, les critères d'évaluation employés, les paramètres expérimentaux ainsi que les choix d'implémentation.

5.4.1.1 Le dataset

Les données utilisées pour appliquer la solution proposée ont été générées par la plateforme réelle LocURa4IoT [140] [141], conçue pour développer et tester des approches de localisation intérieure dans les environnements IoT en se basant sur les mesures ToF. Cette plateforme permet également de tester et évaluer les protocoles de mesure dans les réseaux IoT.
La plateforme repose principalement sur la technologie UWB, reconnue pour sa précision élevée, et intègre également des émetteurs-récepteurs Bluetooth Low Energy (BLE) ainsi que LoRa pour certains noeuds. Pour créer le dataset, plusieurs ancres ont été déployées dans l'environnement IoT considéré. Un objet mobile a é té déplacé à l'intérieur de cet environnement pour collecter des mesures relatives aux signaux échangés entre l'objet et les ancres.
Le protocole de mesure bidirectionnelle (TWR) [38] a été employé lors des échanges de messages entre l'objet mobile (client TWR) et les ancres (serveurs TWR). Le dataset comprend diverses mesures telles que le temps de vol (ToF), les positions des ancres, les positions réelles de l'objet mobile, les mesures de RSSI (Received Signal Strength Indication), les mesures Range, etc. Le Range est une estimation de distance entre deux objets obtenue par un protocole de télémétrie. Cette mesure peut être estimée de la puissance des signaux reçus ou en utilisant le ToF. Les mesures de ToF et de Range ont été calculées en utilisant le protocole TWR [38]. Cependant, le décalage temporel entre l'horloge du serveur TWR et celle du client TWR peut entraîner des mesures de Range inexactes. Ainsi, pour remédier à ce problème, les mesures de Range dans le dataset LocURa4IoT ont été optimisées (en particulier en prenant en compte le skew, c'est-à-dire la différence de fréquence entre les deux noeuds). Ce dataset est accessible en ligne à l'adresse [141].

Pour appliquer la solution de localisation, les entrées du dataset ont été restructurées afin de séparer les paires d'entrées et d'éviter le surajustement. Certains éléments du dataset LocURa4IoT ont été éliminés, notamment :

— Les colonnes contenant des valeurs uniques telles que les identifiants d'objet et le protocole utilisé.

— Les colonnes ne fournissant pas d'informations utiles pour la localisation des objets.

De plus, le dataset a été réorganisé de manière à ce que chaque ligne contienne des informations sur les interactions avec les mêmes 5 ancres. Ensuite, le dataset a été divisé en trois parties : pour l'entraînement, le test et la validation. Dans [150], les auteurs ont considéré le ratio $(p : \sqrt{p} : (\sqrt{p}+1))$ comme étant le ratio de distribution optimal pour ces parties de données. Ici, $p = \sqrt{N}$, où N représente le nombre de lignes uniques dans les données utilisées.
Le nouveau dataset se compose de 3947 lignes uniques. Ainsi, avec $p = 63$, le ratio de distribution est devenu $63 : 8 : 9$. En d'autres termes, le dataset a été divisé comme suit :

— 78.75% des données ont été assignées à l'ensemble d'entraînement

— 10% des données ont été réservées à l'ensemble de validation

— 11.25% des données ont été allouées à l'ensemble de test

5.4.1.2 Les critères d'évaluation

En effet, parmi les mesures d'évaluation disponibles (section 3.3.5 du chapitre 3), les critères suivants ont été sélectionnés pour les expérimentations : MAE, MSE, R2, Accuracy, Recall et F1-score. Le MAE est le critère le plus fréquemment utilisé pour évaluer les solutions de localisation. En outre, dans [151], il a été explicitement recommandé l'utilisation du critère MSE pour résoudre divers problèmes de prédiction, puisqu'il est le plus simple. Les autres sont les critères les plus utilisés pour évaluer les modèles DL.

5.4.1.3 Les paramètres expérimentaux

Dans le cadre des expériences menées, le tableau 5.16 présente les hyperparamètres utilisés pour configurer les modèles DL entraînés et testés.

5.4.1.4 Implémentation

Les expériences ont été réalisées sur un ordinateur doté d'un processeur Intel i7 et de 16 gigaoctets de RAM. Pour mettre en oeuvre l'approche, les choix suivants ont été effectués :

— Utilisation du langage de programmation Python.

TABLEAU 5.16 – Les hyper-paramètres des modèles DL

Hyper-paramètre	Valeur
Nombre d'epochs	2000
Taille du lot (batch size)	32
Taux d'apprentissage	0.001
Nombre de couches cachées	2

— L'IDE Jupyterlab 3.5.1.

— Keras [152] est une bibliothèque Python dédiée au DL. Elle offre une interface de haut niveau avec une abstraction élevée et des Application Programming Interface (API) simples et cohérentes, facilitant ainsi le développement et le test des modèles DL.

— La mise en place de deux optimiseurs mono-objectifs pour l'entraînement des poids d'un modèle DL. Ils se basent sur les algorithmes métaheuristiques suivants :

  — GWO [3] : choisi pour sa récente popularité dans divers domaines [153].

  — AoA [85] : sélectionné pour ses performances supérieures à d'autres algorithmes d'optimisation et ses résultats prometteurs dans la résolution de problèmes complexes [85].

5.4.2 Résultats

Pour évaluer l'efficacité de notre approche développée, nous avons réalisé une série de tests et comparé les modèles créés selon diverses mesures de performance. Comme précisé précédemment, nous avons sélectionné GWO et AoA pour leur récence et leur efficacité parmi les algorithmes d'optimisation métaheuristique mono-objectifs existants. Ils constituent les meilleurs choix pour implémenter notre approche. De plus, le dataset LocURa4IoT offre une variété de mesures permettant de connaître la position exacte des objets mobiles. Ainsi, afin de déterminer quels algorithmes d'optimisation et quelles techniques de mesure à utiliser dans notre solution finale, nous avons effectué des tests et des comparaisons dans les deux sous-sections suivantes.

Tout d'abord, nous avons appliqué notre approche pour développer deux modèles DL de prédiction, utilisant respectivement GWO et AoA et un troisième modèle qui a été développé en utilisant l'algorithme d'optimisation GD. Les résultats de comparaison entre ces trois modèles sont présentés dans la sous-section suivante, démontrant que le modèle basé sur notre approche et sur GWO offre les meilleures performances.

Ensuite, dans la deuxième sous-section, nous avons développé et comparé trois autres modèles de DL, qui sont tous basés sur notre approche et sur GWO, mais utilisant de différentes techniques de mesure pour prédire les positions : ToF, RSSI et Range. Nous avons constaté que les mesures ToF offrent des prédictions de position plus précises que les autres techniques. Par conséquent, notre

proposition finale consiste à entraîner un modèle DL en utilisant notre approche avec GWO et en utilisant les mesures ToF. Cette solution proposée a été comparée à trois autres solutions de localisation existantes parmi les plus récentes et les plus couramment utilisées : une basée sur les réseaux de neurones profonds (DNN), une solution de multilatération et un algorithme Improved Weighted Centroid Localization Algorithm (IWCL). La troisième sous-section fournit des détails sur ces comparaisons et analyse les résultats pour démontrer l'efficacité et les performances supérieures de notre proposition.

5.4.2.1 Comparaison des méthodes d'optimisation

Les performances de trois modèles DL développés ont été évaluées en utilisant les mêmes hyperparamètres (voir tableau 5.16). Ces modèles sont basés principalement basés sur les mesures ToF du dataset LocURa4IoT. Ces données incluent les coordonnées des ancres et les mesures ToF. Deux modèles ont été entraînés en appliquant la méthode proposée (voir section 4.5 du chapitre précédent) avec GWO et AoA respectivement. Le troisième modèle a été entraîné en utilisant une méthode d'optimisation existante (GD).
La figure 5.15 présente les courbes d'apprentissage des trois modèles étudiés, illustrant la variation des valeurs de MAE au fil des epochs d'entraînement avec les trois optimiseurs. Chaque deux courbes sont tracées ensemble pour faciliter la comparaison entre les modèles : (i) la figure 5.15(a) compare les deux modèles DL développés avec notre approche utilisant GWO et AoA, (ii) la figure 5.15(b) compare les modèles développés avec GWO et GD respectivement, et (iii) la figure 5.15(c) compare le modèle développé avec AoA à celui développé avec GD. Il est important de noter que, tout au long des epochs d'entraînement, les paramètres ont été ajustés pour trouver les valeurs optimales conduisant à des MAE plus faibles. Cette figure montre que l'optimisation des paramètres entraînables avec notre approche, utilisant GWO, a conduit à la convergence la plus rapide par rapport aux deux autres optimiseurs. En d'autres termes, les erreurs de MAE étaient plus petites et diminuaient plus rapidement lorsque les équations de GWO étaient utilisées pour mettre à jour les valeurs de poids lors des itérations d'apprentissage. De plus, cette figure indique que l'optimisation avec la méta-heuristique AoA est moins efficace que celle réalisée avec GD.
Ainsi, d'après ces courbes, nous pouvons déduire que notre approche utilisant GWO surpasse les deux autres méthodes, suivie par l'optimiseur GD. Cependant, l'optimisation avec AoA se révèle être la moins efficace. Cette conclusion est également confirmée par la figure 5.16. Dans celle-ci, les valeurs moyennes des erreurs MAE (figure 5.16(a)) et MSE (figure 5.16(b)) à travers tous les epochs lors de la phase d'entraînement sont comparées. Cette analyse confirme que l'entraînement avec GWO dépasse celui effectué par les deux autres optimiseurs (GD et AoA) et que notre approche basée sur GWO accélère la convergence vers les paramètres optimaux et améliore l'apprentissage du modèle DL.

En outre, les temps d'apprentissage des trois modèles sont comparés dans le tableau 5.17. En termes de rapidité d'entraînement, l'optimisation avec GD est la plus efficace, suivie de celle avec GWO, puis avec AoA. Cependant, ce critère de temps d'apprentissage n'est pas essentiel car il concerne la phase hors ligne (entraînement). Pendant l'exécution des applications IoT (phase en ligne), les objets utilisent un modèle pré-entraîné pour se localiser.

FIGURE 5.15 – Courbes d'apprentissage pour les modèles DL optimisés avec la solution proposée (utilisant GWO et AoA) et GD

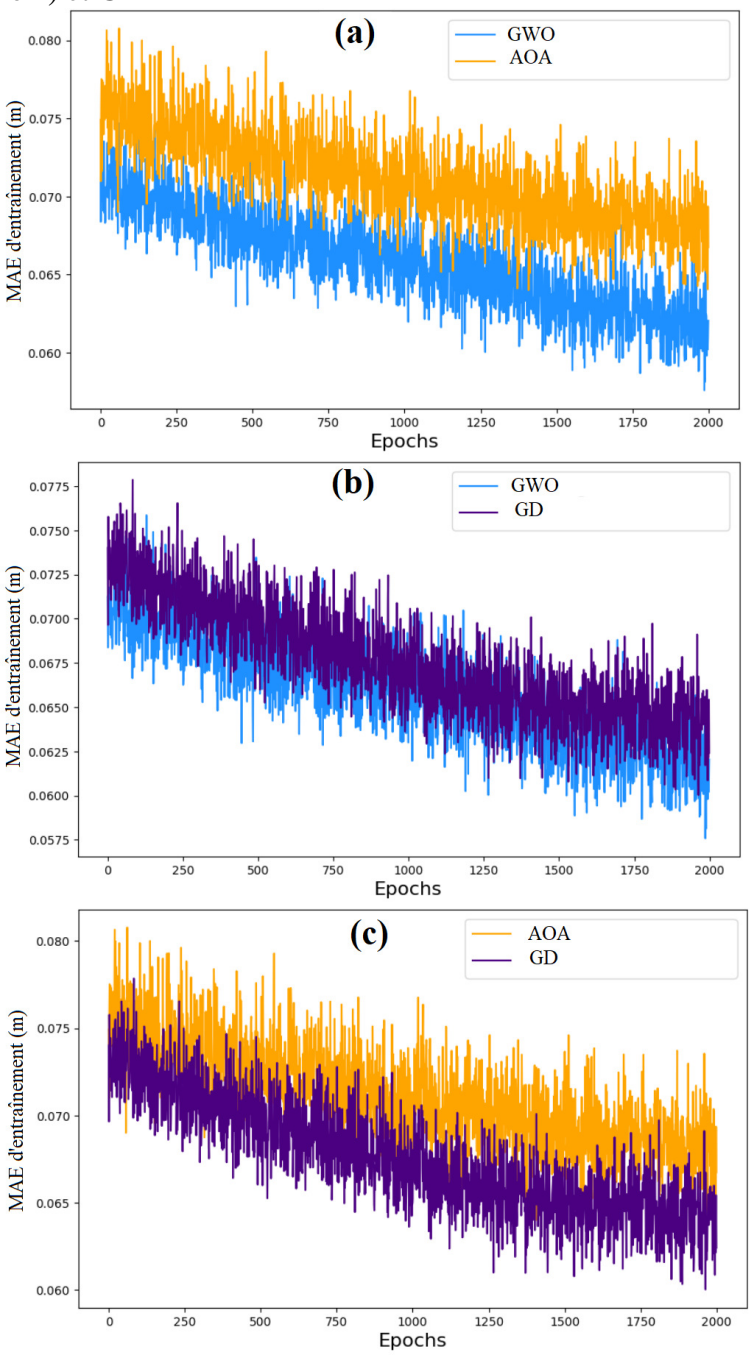
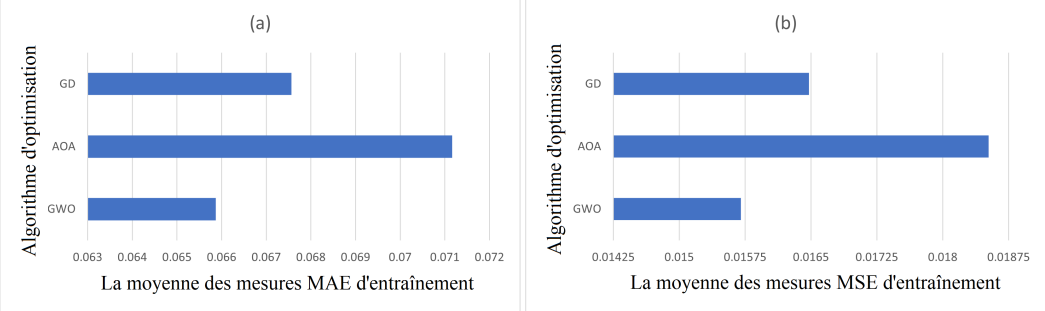


FIGURE 5.16 – Comparaison des valeurs moyennes globales des erreurs MAE et MSE à travers touts les epochs d'entraînement pour les modèles développés avec GWO, AoA et GD



Ainsi, basé sur ces résultats, l'optimisation utilisant l'approche proposée avec GWO a été sélectionnée pour mener les tests restants présentés dans cette section.

TABLEAU 5.17 – Comparaison des temps d'entraînement

Algorithme d'optimisation	**Temps d'entraînement (secondes)**
GWO	328.302
AoA	393.162
GD	281.52

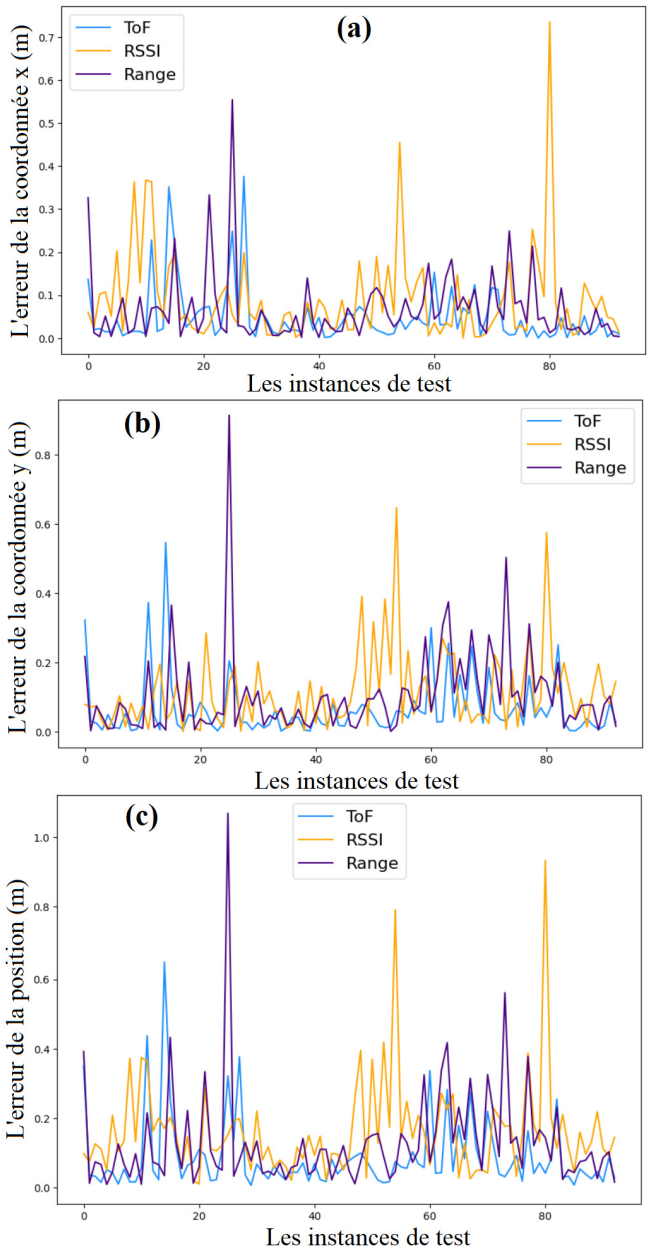
5.4.2.2 Comparaison des techniques de mesure

En fait, les tests initiaux ont démontré que l'entraînement basé sur l'approche développée avec GWO donne les meilleurs résultats. Par conséquent, cette méthode d'optimisation a été utilisée pour entraîner trois modèles DL avec les mêmes hyperparamètres, mais en utilisant différentes mesures sélectionnées parmi celles proposées par LocURa4IoT. En plus des positions des ancres, les données utilisées comprennent respectivement les mesures de temps de vol (ToF), de la force du signal reçu (RSSI) et de Range. Cette étape de test visait à comparer les performances des trois modèles développés en termes de précision de prédiction de position. Les modèles entraînés ont été utilisés pour prédire les positions en utilisant les données de test (11.25 % du dataset). Les coordonnées x et y prédites, qui ont été obtenues à partir des modèles ont été comparées, et ces prédictions ont été utilisées pour calculer la distance entre la position réelle de l'objet mobile et sa position estimée. En fait, la figure 5.17 présente une étude comparative des résultats obtenus par les trois modèles. Elle montre les courbes de variation des erreurs pour les coordonnées x (figure 5.17(a)) et y (figure 5.17(b)), ainsi que la distance entre les positions réelles et prédites (figure 5.17(c)). Ces courbes montrent que les erreurs obtenues à partir de ToF UWB sont généralement plus faibles que celles obtenues avec les mesures de RSSI et de Range.
La figure 5.18 confirme que les prédictions de positions basées sur les mesures de ToF UWB sont plus précises que celles obtenues avec le RSSI et le Range. Les sous-figures (a), (c) et (e) de la

figure 5.18 comparent les valeurs MAE calculées à partir des prédictions pour les coordonnées x et y, ainsi que les positions (mesurées comme les distances entre les positions réelles et estimées). En outre, les sous-figures (b), (d) et (f) présentent des comparaisons des valeurs MSE pour les prédictions en termes de coordonnées x et y ainsi que les positions.

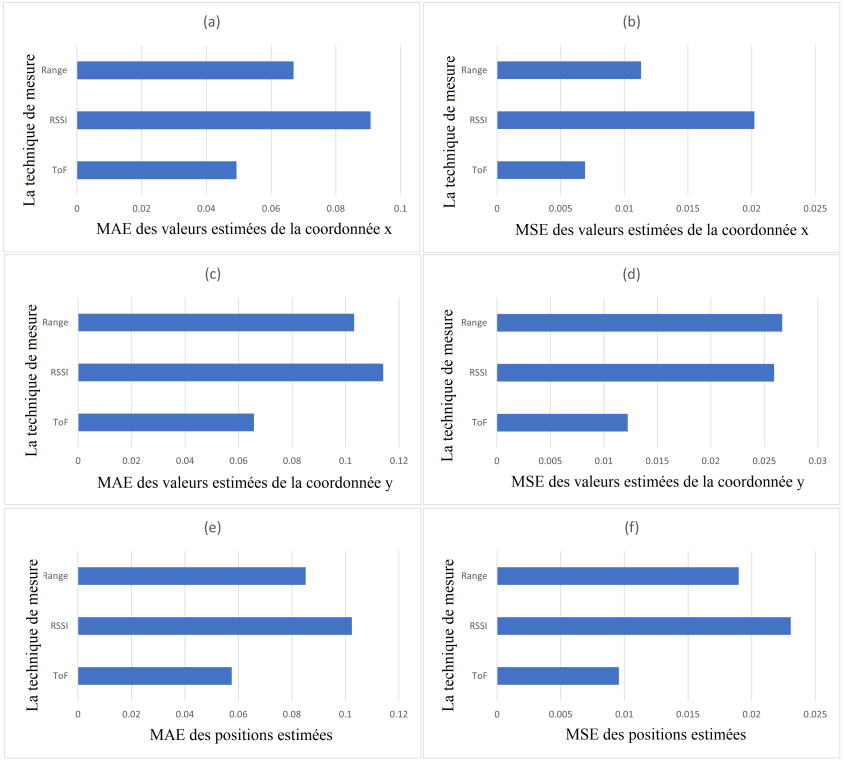
Nous remarquons que les mesures de ToF ont produit les valeurs les plus faibles des critères MAE et MSE. De plus, il est évident que les prédictions basées sur le Range ont entraîné des valeurs

FIGURE 5.17 – Courbes de variation des erreurs obtenues à partir des prédictions utilisant les mesures de ToF, RSSI et Range en termes de coordonnées x et y ainsi que la distance entre les positions réelles et prédites



MAE et MSE plus petites que le RSSI pour la coordonnée x et la position. Cependant, pour la coordonnée y, le Range a donné une valeur MAE plus faible et une valeur MSE légèrement plus élevée que le RSSI.

FIGURE 5.18 – 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 techniques de mesure comparées



Ainsi, en nous basant sur ces résultats, nous pouvons dire que l'optimisation utilisant l'approche proposée avec GWO et l'intégration des mesures UWB ToF a été sélectionnée comme la solution de localisation en intérieur proposée par cette étude, en raison de ses performances supérieures par rapport aux autres solutions développées et testées. Par conséquent, elle sera comparée à d'autres solutions de localisation dans la sous-section suivante.

5.4.2.3 Comparaison des performances de la solution proposée avec la multilatération, l'algorithme IWCL et une solution de DL

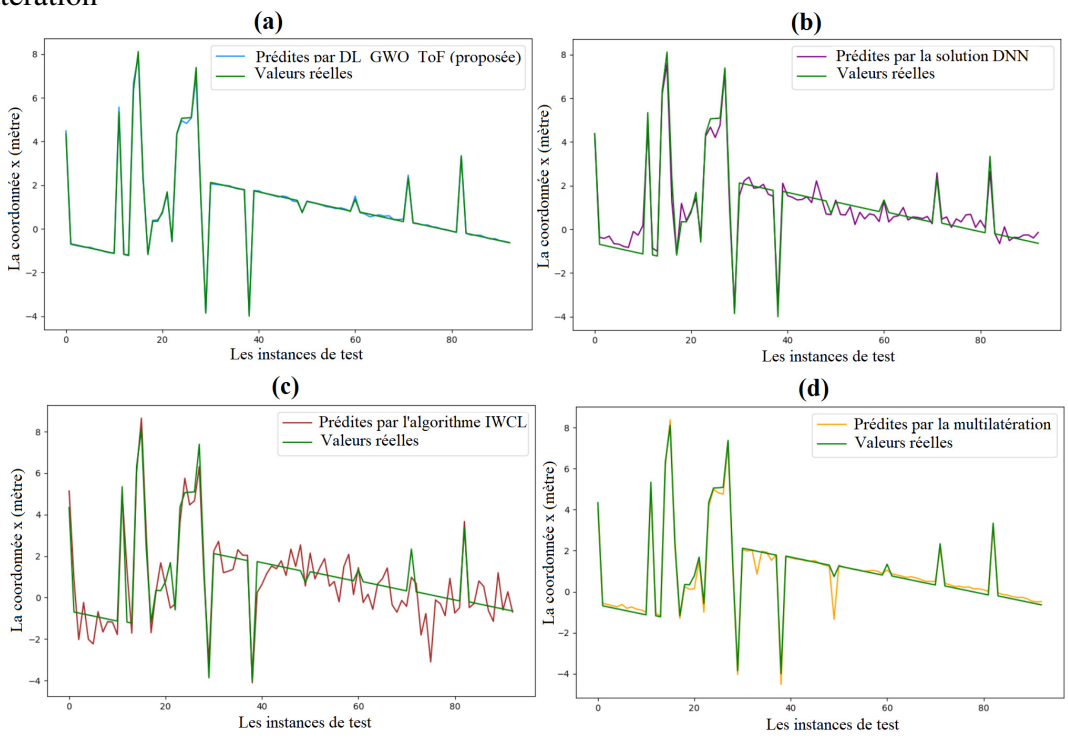
Cette section de test vise à comparer les performances de notre solution, Deep Learning-Gray Wolf Optimizer-Time of Flight localization solution (DL-GWO-ToF), avec trois autres méthodes de localisation couramment utilisées : une méthode basée sur le DL [154], le IWCL [155], et une méthode de multilatération présentée dans [156]. L'objectif est de démontrer que notre approche

offre des performances supérieures. Les capacités d'estimation de position de chaque méthode sont évaluées en utilisant les mêmes données de test.

a) Comparaisons standards

Les résultats d'estimation des 4 solutions de localisation sont comparés dans les figures 5.19 et 5.21.

FIGURE 5.19 – Comparaison des valeurs réelles de la coordonnée x et des valeurs estimées obtenues en appliquant la solution proposée, la solution DNN, l'algorithme ICWL et la méthode de multilatération

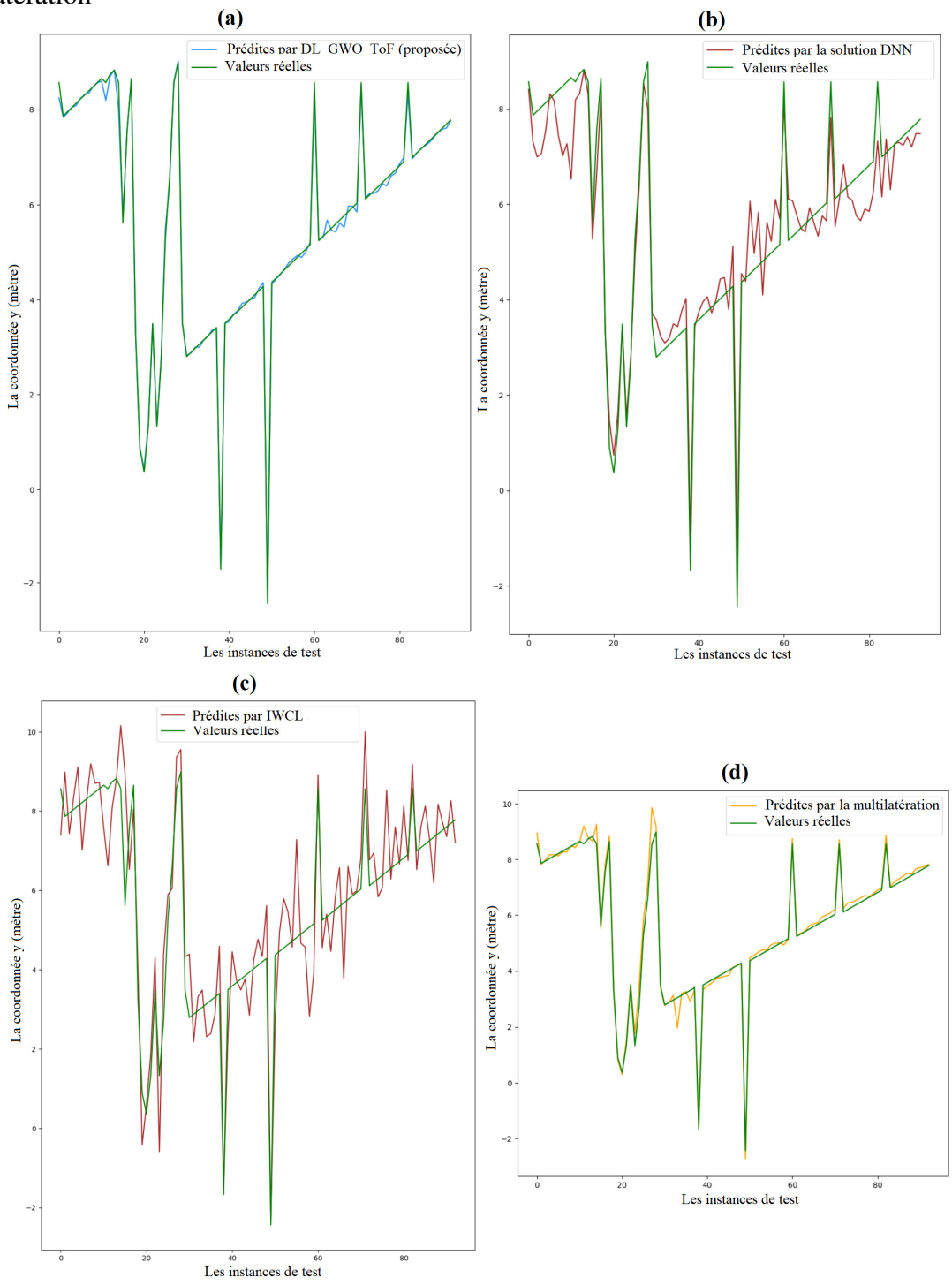


Les figures 5.19 et 5.20 comparent les valeurs prédites et réelles des coordonnées x et y pour les différentes solutions étudiées. Les courbes indiquent que les prédictions effectuées par l'approche présentée sont plus précises que celles des autres méthodes. Notamment, les sous-figures 5.19(a) et 5.20(a) montrent que les prédictions obtenues avec notre méthode correspondent presque parfaitement aux valeurs réelles des coordonnées x et y. En revanche, les valeurs prédites par la solution DNN sont représentées dans les sous-figures 5.19(b) et 5.20(b). Les résultats les plus proches en termes de performance sont obtenus par la multilatération (sous-figures 5.19(c) et 5.20(c)). Par contre, les prédictions des coordonnées x et y par l'algorithme IWCL, comme illustré dans les sous-figures 5.19(d) et 5.20(d), sont les plus éloignées des valeurs réelles.

La figure 5.21 présente une étude comparative des valeurs MAE et MSE calculées à partir des prédictions obtenues par les solutions comparées. Les sous-figures 5.21(a) et 5.21(b) représentent

les valeurs MAE et MSE obtenues en prédisant la coordonnée x, tandis que les valeurs prédites de la coordonnée y sont représentées dans les sous-figures 5.21(c) et 5.21(d). De plus, les valeurs MAE

FIGURE 5.20 – Comparaison des valeurs réelles de la coordonnée y et des valeurs estimées obtenues en appliquant la solution proposée, la solution DNN, l'algorithme ICWL et la méthode de multilatération



et MSE des distances entre les positions réelles et estimées sont présentées dans les sous-figures 5.21(e) et 5.21(f).

En effet, ces résultats montrent que l'approche développée a fourni des erreurs MAE et MSE plus faibles pour les trois mesures considérées (x, y et position). En termes de valeurs MAE pour la prédiction de la coordonnée x, la solution proposée se distingue avec le plus faible MAE de 0.0492, ce qui montre sa précision dans la prédiction de la coordonnée x. La multilatération suit avec un MAE plus élevé, et la solution DNN et l'algorithme IWCL ont des valeurs MAE considérablement plus élevées, indiquant des prédictions de coordonnées x moins précises. Cela met en évidence les performances supérieures de la solution proposée dans l'estimation précise de la coordonnée x.

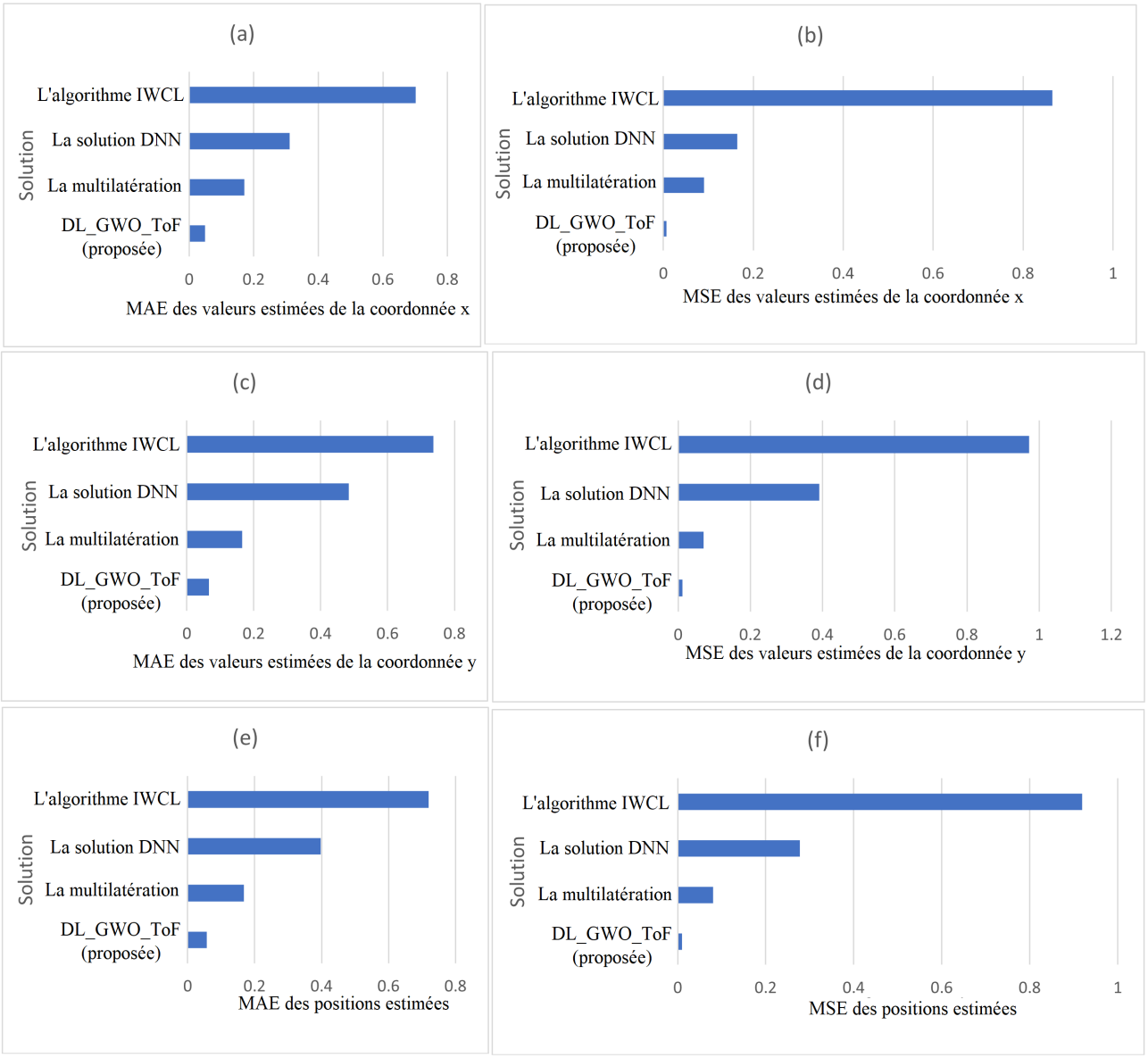
Une tendance similaire est observée dans les valeurs MAE pour la prédiction de la coordonnée y. La solution proposée maintient le MAE le plus bas, suivi par la multilatération, tandis que la solution DNN et l'algorithme IWCL présentent des valeurs MAE plus élevées. Cela souligne la précision supérieure de l'approche proposée dans la prédiction de la coordonnée y. Les valeurs MAE globales de prédiction de position mettent en évidence que la proposition surpasse les autres solutions, car elle fournit les estimations de position les plus précises. En examinant les valeurs MSE, la proposition surpasse les autres solutions, notamment en termes de prédictions des coordonnées x et y. 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%. Ces résultats démontrent collectivement l'efficacité de la solution proposée dans l'obtention d'une localisation intérieure précise, ce qui est crucial pour les applications IoT.

De plus, l'approche présentée a fourni une erreur moyenne entre les positions réelles et prédites de 0.057m. Ces résultats démontrent également que la méthode proposée a surpassé les solutions de localisation récentes, décrites dans la section 2.4.2, où la meilleure erreur de distance moyenne était de quelques dizaines de centimètres.

En outre, le tableau 5.18 offre une analyse exhaustive de différentes solutions de localisation, évaluées selon plusieurs critères de performance clés : la précision de la localisation, l'erreur moyenne de position, le coefficient de détermination R2, le rappel, le score F1, ainsi que les temps nécessaires pour appliquer ces méthodes. Outre les MAE et MSE, le coefficient de détermination R2 peut généralement évaluer la performance du modèle pour les problèmes de prédiction. Cependant, la précision, le rappel et le score F1 sont des mesures utilisées pour les problèmes de classification, impliquant la classification des points de données en catégories. Dans le cas de la régression, notre objectif est plutôt de prédire une valeur numérique continue, et non de catégoriser les données en classes. C'est pourquoi une version personnalisée de ces métriques est élaborée pour évaluer le problème de localisation. En pratique, la précision, le rappel et le score F1 sont calculés pour la régression en considérant les valeurs dans une marge de tolérance comme des vrais positifs. Étant donné que le choix de cette marge peut avoir un impact significatif sur les résultats, sa valeur est

généralement définie en fonction des exigences de l'application pour laquelle la solution de localisation est destinée. Dans notre étude, nous utilisons un seuil de 0.5 mètre comme exemple. En d'autres termes, une prédiction est considérée comme réussie uniquement si la différence entre les valeurs réelles et prédites est inférieure à 0.5 mètre. Par conséquent, dans ce contexte, la précision, le rappel et le score F1 sont utilisés pour mesurer dans quelle mesure les solutions identifient correctement les positions. De plus, dans le tableau 5.18, la colonne "Temps d'entraînement" indique le temps nécessaire pour entraîner les modèles de localisation. Cette mesure n'est pas applicable

FIGURE 5.21 – 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



pour les solutions IWCL et de multilatération. La colonne "Temps d'estimation" reflète le temps nécessaire pour effectuer les prédictions ou estimer les positions.

Notre solution proposée a surpassé la méthode de multilatération en atteignant une précision de localisation de 98.92%, tandis que cette dernière a obtenu une précision de 89.24% (avec un seuil de 0.5 mètre). Parmi les solutions comparées dans le tableau 5.18, 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.

En revanche, l'algorithme IWCL affiche la plus faible précision de localisation à 19.35% avec une erreur de position moyenne élevée de 0.719 mètre. De plus, il présente un score R2 limité à 0.817, suggérant une prédictibilité réduite. Ses scores de rappel (0.120) et de score F1 (0.562) sont également les plus bas, indiquant des difficultés à identifier précisément les positions. Cependant, son temps d'estimation est rapide à 0.064 seconde, ce qui le rend potentiellement adapté aux applications en temps réel.

La méthode de multilatération offre une estimation relativement rapide en 1.982 secondes avec une précision de 89.24% et une erreur de position moyenne de 0.167 mètre. Elle présente un bon rappel (0.805) et un bon score F1 (0.969), indiquant sa capacité à identifier correctement les positions avec un certain compromis entre précision et rappel.

Enfin, la solution basée sur les réseaux de neurones (DNN) présente une précision de localisation inférieure à 51.61%, avec des scores de rappel (0.377) et de score F1 (0.829) relativement plus bas. Bien que sa précision soit modérée, elle offre des temps d'entraînement et d'estimation comparables à la solution proposée.

En outre, le temps requis pour prédire la position en utilisant l'approche introduite était de 0.155 seconde, tandis que la solution de multilatération nécessitait 1.982 secondes pour estimer les coordonnées. Bien que la solution développée ait surpassé l'autre méthode en termes de temps d'estimation, il est important de noter que la solution DL proposée nécessitait une phase préliminaire hors ligne, demandant 328.302 secondes pour entraîner le modèle de prédiction. D'autre part, la méthode de multilatération ne nécessitait aucune étape supplémentaire en dehors de l'estimation de position en temps réel.

b) Tests statistiques inférentiels

Pour évaluer si la solution proposée représente une amélioration significative par rapport aux autres

TABLEAU 5.18 – 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

TABLEAU 5.19 – Les rangs moyens des solutions de localisation et le calcul des statistiques de Friedman et d'Iman-Davenport pour les erreurs de position

Solution	DL-GWO-ToF (proposée)	Multilatération	Solution DNN	Algorithme IWCL
Rang Moyen	**1.45**	2.77	5.41	9.88
Statistique de Friedman = 9.53 Statistique d'Iman-Davenport = 44.53 F(3,276) = 2.08				

méthodes de localisation, nous avons recouru à des tests statistiques inférentiels, comme exposé dans cette sous-section. En fait, les tests statistiques sont couramment utilisés pour évaluer la performance relative d'un nouvel algorithme par rapport aux solutions existantes pour un problème spécifique. De ce fait, étant donné que notre objectif est d'évaluer la solution proposée par rapport aux alternatives existantes, nous avons effectué des tests impliquant des comparaisons (1 x N). Pour cela, nous avons utilisé le test de Friedman avec l'extension d'Iman-Davenport et la procédure post-hoc de Holm, conformément aux recommandations de [149]. Les paragraphes suivants détaillent les étapes de mise en oeuvre de ces deux méthodes.

— **Application du test de Friedman avec l'extension d'Iman-Davenport** :

Le test de Friedman avec l'extension d'Iman-Davenport permet de comparer un algorithme à une collection d'autres algorithmes et d'identifier les différences statistiquement significatives entre eux. Dans ce contexte, il est utilisé pour évaluer la solution proposée par rapport aux autres solutions, en termes d'erreurs de position (distance entre les positions réelles et prédites). Les étapes de l'application de ce test sont détaillées dans la section 5.3.3.1.

Le tableau 5.19 présente les rangs moyens obtenus par les solutions de localisation comparées pour toutes les instances de test en termes de métrique d'erreur de position. Le rang le plus favorable (indiquant une performance supérieure) est mis en gras, et le rang suivant est souligné. De plus, le tableau 5.19 fournit les résultats statistiques des tests de Friedman et d'Iman-Davenport, ainsi que leurs valeurs associées dans la table de distribution F.

L'analyse des résultats fournis montre des différences significatives dans les rangs moyens de performance parmi les solutions de localisation comparées. La solution proposée a obtenu le meilleur rang moyen de 1.45, prouvant sa précision supérieure dans la prédiction des posi-

tions. La multilatération démontre également une performance solide avec un rang moyen de 2.77. En revanche, la solution basée sur les réseaux de neurones (DNN) et l'algorithme IWCL ont des rangs moyens plus élevés de 5.41 et 9.88, respectivement. Ces résultats indiquent que l'approche DL-GWO-ToF proposée surpasse les autres alternatives, et que la multilatération offre également une performance fiable.

D'autre part, les statistiques de Friedman et d'Iman-Davenport fournissent des informations supplémentaires sur la signification de ces différences. En fait, la statistique de Friedman, avec une valeur de 9.53, montre qu'il existe des variations significatives dans les rangs de performance des solutions. Cela correspond aux différences observées dans les rangs moyens. La statistique d'Iman-Davenport, égale à 44.53, complète le test de Friedman en mettant en évidence les disparités entre les solutions. Lorsqu'elle est comparée à la valeur critique de la table de distribution F (F(3.276) = 2.08), qui sert de seuil, la statistique de Friedman obtenue de 9.53 dépasse ce seuil. Ce résultat appuie la conclusion selon laquelle il existe une distinction statistiquement significative dans la performance parmi les solutions de localisation.

— **Application de la procédure post-hoc de Holm** : Les étapes impliquées dans l'application de la procédure post-hoc de Holm sont détaillées dans la section 5.3.3.2.

TABLEAU 5.20 – Résultats du test post-hoc de Holm pour la comparaison entre les solutions de localisation en termes d'erreurs de position

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

Le tableau 5.20 résume les résultats de la procédure post-hoc de Holm appliquée à l'indicateur d'erreur de position. Il est notable que toutes les valeurs z obtenues sont négatives, confirmant ainsi que la solution proposée DL-GWO-ToF surpasse de manière constante toutes les autres solutions de localisation comparées.

L'analyse comparative des solutions de localisation à l'aide du test post-hoc de Holm met en évidence la performance supérieure de la solution DL-GWO-ToF. DL-GWO-ToF est considérée comme la référence par rapport à laquelle les autres solutions sont évaluées. De plus, sa valeur de p corrigée par Holm de 0.0125 confirme sa supériorité et montre que la solution DL-GWO-ToF surpasse toutes les autres méthodes étudiées.

En revanche, les autres solutions, la multi-latération, la solution DNN et l'algorithme IWCL, présentent des performances nettement inférieures par rapport à DL-GWO-ToF. Leurs valeurs z négatives, notamment celles de la solution DNN et de l'algorithme IWCL, avec respectivement -7.44 et -14.62, soulignent leur performance significativement moindre. Ces ré-

sultats, combinés à leurs rangs inférieurs et à leurs valeurs de p corrigées par Holm très faibles, indiquent que ces solutions sont bien moins performantes que DL-GWO-ToF. Dans l'ensemble, les analyses statistiques inférentielles confirment la supériorité de la solution DL-GWO-ToF par rapport aux autres méthodes de localisation évaluées.

5.4.2.4 Synthèse

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 ce travail, 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 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 ré-entraînement 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].

5.5 Résultats expérimentaux du routage géographique basé sur IMOGWO

La présente section est dédiée à la présentation et à l'analyse des résultats des expérimentations menées afin de tester la contribution du routage géographique décrit dans la section 4.7. L'objectif

principal de ces expérimentations est triple : évaluer les performances de la solution de routage géographique et hybride proposée, comparer ces performances avec celles d'autres solutions de routage existantes, et tester l'efficacité de l'approche IMOGWO présentée dans la section 4.4, en l'appliquant à un problème réel autre que les problèmes de référence DTLZ. Ces analyses et comparaisons permettront d'appréhender les avantages et les limitations de la méthode proposée dans un contexte pratique.

5.5.1 Expérimentations par simulations hybrides

Nous avons mis en oeuvre des simulations hybrides, intégrant à la fois des éléments simulés et des éléments réels. L'intégration des objets réels dans ces simulations permet de se rapprocher au mieux des conditions et des environnements réels. Cette approche hybride permet d'obtenir des résultats plus proches de la réalité tout en conservant la flexibilité et la précision des simulations virtuelles.

5.5.1.1 Le simulateur IoT

Nous avons choisi d'utiliser le simulateur IoT CupCarbon [157][158] pour réaliser les simulations hybrides. En effet, CupCarbon est un outil largement utilisé dans le domaine de la recherche pour sa capacité à modéliser et simuler des réseaux d'objets connectés. Son principal avantage réside dans sa capacité à intégrer à la fois des objets réels et des objets simulés, chacun présentant une diversité de caractéristiques, le tout au sein d'une simulation hybride. Les objets simulés sont codés en utilisant le langage Python. Les objets réels et les objets simulés communiquent en utilisant le protocole Message Queuing Telemetry Transport (MQTT) [159].

5.5.1.2 Les objets réels M5StickC

Dans le cadre de nos simulations hybrides, nous avons choisi d'intégrer les objets réels M5StickC [143]. Ils offrent une multitude de fonctionnalités pour la collecte de données en temps réel et sont largement utilisés dans divers domaines de l'IoT. Leur taille compacte et leur polyvalence en font un choix idéal pour nos expérimentations, permettant une intégration fluide dans nos simulations hybrides.

5.5.1.3 Les paramètres et la configuration des expérimentations

Le tableau 5.21 présente les paramètres utilisés lors des expérimentations. Le nombre de cycles exécutés est 500. Un cycle désigne une communication de bout en bout qui se produit entre 2 objets du réseau pour émettre un message, nécessitant ainsi l'établissement d'une route multi-sauts

pour échanger les données. En ce qui concerne le nombre d'objets, nous avons inclus un total de 100, parmi lesquels 90 sont simulés : 70 objets fixes et 20 objets mobiles, tandis que 10 sont des dispositifs réels M5StickC. Les objets mobiles suivent des trajectoires aléatoires. En ce qui concerne l'énergie initiale, les objets simulés ont une énergie initiale variable puisque nous visons considérer à des objets hétérogènes en termes de capacité énergétique. En revanche, les dispositifs réels M5StickC sont équipés de batteries de 95 mAh sous 3.7V. Les énergies de transmission d'un bit, (E_{Tx_bit}), et de réception d'un bit, (E_{Rx_bit}), sont également mentionnées dans le tableau. La technologie de communication utilisée est le Wi-Fi. Ce réseau Wi-Fi est supposé en mode ad hoc, autorisant ainsi une communication directe entre les divers noeuds sans dépendre d'une infrastructure centralisée.
La figure 5.23 illustre le modèle de la simulation hybride effectuée avec CupCarbon, où les liaisons de communication ne sont pas affichées. Les objets sont répartis de façon aléatoire dans le réseau. En revanche, la figure 5.24 présente le même modèle avec l'affichage des liaisons de communication.

TABLEAU 5.21 – Les paramètres des expérimentations

Paramètre	**Valeur**	
Nombre de cycles	500	
Nombre d'objets	total	100
	simulés fixes	70
	simulés mobiles	20
	réels M5StickC	10
Mobilité	aléatoire	
Taille initiale du message	64-80 bits	
Énergie initiale	objets simulés	variable
	objets réels M5StickC	95 mAh à 3.7V
E_{Tx}	objets simulés	10 nJ/bit
E_{Rx}	objets simulés	5 nJ/bit

FIGURE 5.22 – Les objets IoT mini-ESP32 M5StickC utilisés dans les expérimentations

5.5.2 Les critères d'évaluation

Dans ce cadre, nous avons sélectionné des critères d'évaluation afin d'évaluer notre approche proposée et la comparer avec d'autres solutions existantes. Les indicateurs de performances utilisés sont :

— La latence de transmission : elle consiste au temps écoulé entre l'émission d'un paquet de données par la source initiale et sa réception par la destination finale.

FIGURE 5.23 – Le modèle de la simulation hybride sans affichage des liens de communications

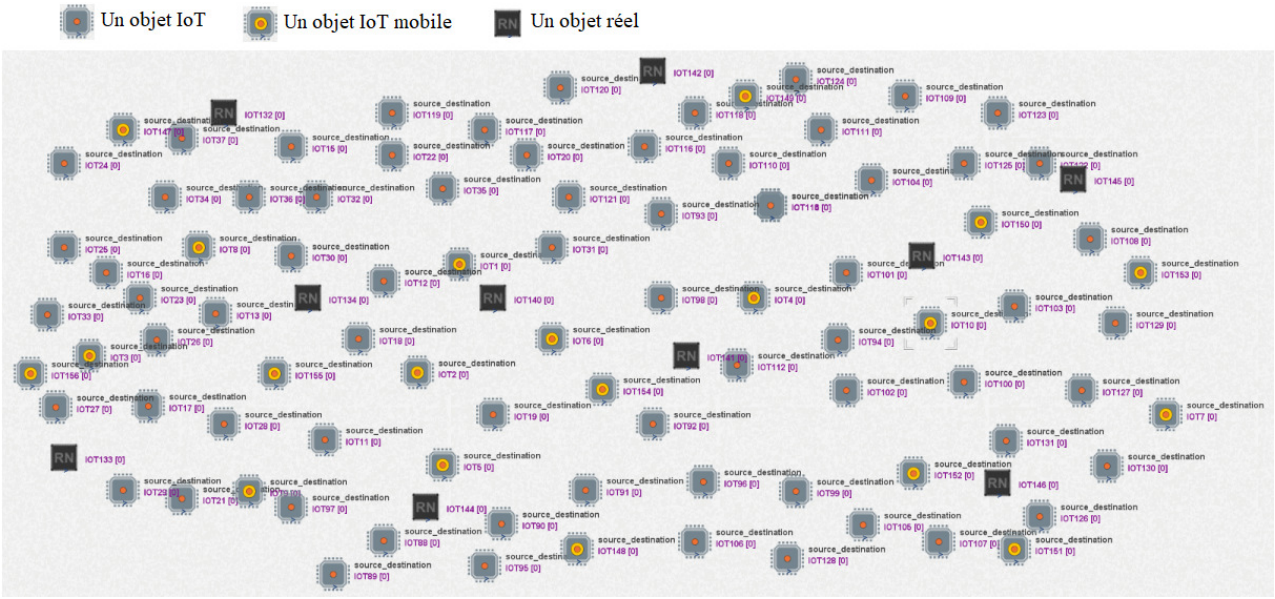
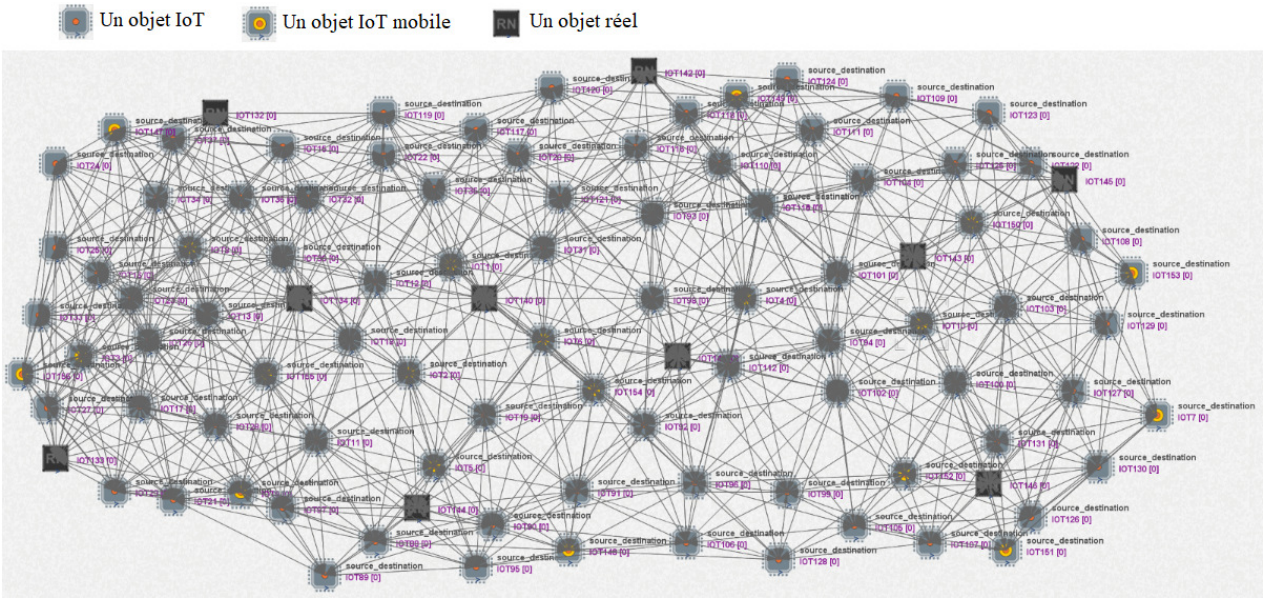


FIGURE 5.24 – Le modèle de la simulation hybride avec affichage des liens de communications



— Le PDR : il désigne la proportion de paquets de données émis qui parviennent à destination avec succès (équation 5.6). Elle fait référence à la capacité du protocole de routage à livrer correctement les paquets de données à leur destination sans perte ni corruption.

$$PDR = \frac{Nombre\_de\_paquets\_correctement\_reçus}{Nombre\_total\_de\_paquets\_émis} \times 100\% \tag{5.6}$$

— Le nombre d'objets actifs durant les cycles exécutés.

— Le First Node Dies (FND) : il désigne la période qui sépare le début des opérations de communication dans le réseau et le moment où le premier noeud épuise complètement son énergie et cesse de fonctionner. Il fait référence à la période de stabilité du réseau.

— L'overhead de contrôle : il s'agit de la proportion de la quantité de données de contrôle supplémentaire utilisées par les messages de contrôle de routage. Un faible overhead de contrôle est souhaitable pour minimiser l'impact sur les performances du réseau. L'overhead de contrôle peut être calculé selon l'équation suivante :

$$Overhead\_de\_contrôle = \frac{Taille\_totale\_des\_messages\_de\_contrôle}{Taille\_totale\_des\_messages\_des\_donnes\_transmises} \times 100\% \tag{5.7}$$

Nous avons choisi ces critères de performances en raison de leur large utilisation et de leur capacité à évaluer les différents aspects de performance des solutions de routage étudiées, incluant l'efficacité, la fiabilité, l'optimisation d'énergie, l'équilibrage de charge et la stabilité.

5.5.3 Les algorithmes de routage comparés

En effet, dans l'analyse comparative, nous avons comparé les résultats des solutions de routage suivantes :

— Notre approche de routage géographique et hybride décrite dans la section 4.7 développée avec notre proposition IMOGWO (section 4.4) ;

— Notre approche de routage géographique et hybride décrite dans la section 4.7 développée avec MOGWO [2] ;

— Le routage basé sur la QoS proposé et présenté dans [83] ;

— Le routage géographique proposé et présenté dans [109].

En premier lieu, notre approche de routage géographique et hybride a été développée en deux versions, en appliquant respectivement IMOGWO et MOGWO. Puisque le simulateur peut fournir les positions des objets, les positions considérées lors du routage géographique sont calculées à partir

de ces positions fournies, en tenant compte de l'erreur moyenne obtenue par notre solution de localisation proposée dans la section précédente, qui est de 0.057 mètres. L'objectif de la comparaison des résultats obtenus par les deux algorithmes d'optimisation est de mettre en évidence la capacité de notre proposition IMOGWO à produire des résultats meilleurs que MOGWO dans la résolution d'un problème réel de routage, en dehors des problèmes de référence DTLZ. En deuxième lieu, nous avons sélectionné les deux solutions de routage proposés respectivement dans [83] et [109] afin de comparer les résultats obtenus avec notre approche. En fait, ce choix est justifié par le fait qu'ils sont parmi les solutions de routage, basées sur l'optimisation métaheuristique, récemment proposées (voir tableau 3.3) et qu'ils ont démontré des performances prometteuses. L'algorithme de routage décrit dans [83] s'appuie sur AcNSGA-III, un algorithme d'optimisation métaheuristique proposé. Les évaluations menées pour le tester ont utilisé des objets M5StickC. L'algorithme de routage décrit dans[109] est de même nature géographique que notre proposition. Il est basé sur l'algorithme d'optimisation métaheuristique BFO.
Les algorithmes de routage comparés sont tous évalués dans les mêmes conditions et le même environnement initial. Le tableau 5.22 contient les valeurs des paramètres des métaheuristiques, AcNSGA-III et BFO, appliquées pour développer ces algorithmes.

TABLEAU 5.22 – Les paramètres des algorithmes AcNSGA-III et BFO

Paramètre	**Valeur**	
Taille de la population	100	
Nombre de générations	500	
AcNSGA-III	Indice de recombinaison	0.8
	Opérateur de recombinaison	SBX
	Probabilité de recombinaison	20
	Indice de mutation	50
	Opérateur de mutation	Bit-flip
	Probabilité de mutation	1/450
BFO	Étapes Swim	4
	Étapes chemotactic	100
	Étapes de reproduction	5
	Étapes d'élimination et de dispersion	2
	Probabilité d'élimination	Bit-flip
	L'unité de run-length	$10^{-3} \times R$

5.5.4 Résultats et discussion

Les résultats des algorithmes de routage sont comparés à travers deux approches : des comparaisons standards et des tests statistiques inférentiels. Les comparaisons standards permettent une évaluation qualitative des performances des différents algorithmes en examinant directement les résultats obtenus. De plus, nous avons utilisé des tests statistiques inférentiels pour une analyse plus appro-

fondie permettant de déterminer si les différences observées entre les performances des algorithmes sont statistiquement significatives.

5.5.4.1 Les comparaisons standards

La figure 5.25 illustre la variation du nombre d'objets actifs pendant les cycles réalisés pour les algorithmes de routage comparés. Un objet devient inactif lorsqu'il épuise son énergie, ce qui est représenté par une diminution du nombre d'objets actifs au fil des cycles. Les résultats montrent que notre approche de routage géographique avec IMOGWO et celle avec MOGWO se distinguent par leur capacité à maintenir un nombre d'objets actifs relativement constant sur une grande partie des cycles de simulation, suggérant ainsi une bonne répartition initiale de la charge énergétique. Cependant, l'approche IMOGWO semble offrir une meilleure stabilité à long terme, avec une diminution progressive du nombre d'objets actifs à partir d'un certain point, tandis que MOGWO montre une tendance à une diminution plus rapide. D'autre part, bien que les algorithmes AcNSGA-III et Bacteria Foraging Optimization Algorithm (BFOA) commencent avec une répartition initiale de la charge énergétique similaire aux approches de routage géographique, ils semblent moins efficaces dans la gestion de la consommation énergétique, avec une diminution plus rapide du nombre d'objets actifs au fil du temps.
En fait, ces résultats sont confirmés par la figure 5.26 qui compare les valeurs du critère FND pour les algorithmes de routage. Le FND désigne le moment où le premier objet épuise entièrement son énergie. En d'autres termes, la figure 5.26 compare les périodes de stabilité offertes par les solutions de routage étudiées. Ainsi, 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.
La figure 5.27 compare les valeurs de la latence moyenne de transmission obtenues par les algorithmes de routage pendant les cycles effectués. Chaque valeur représentée dans la figure correspond à la latence moyenne évaluée sur une période de 100 cycles successifs. De plus, la figure 5.28 compare les valeurs de la latence moyenne de transmission pour l'ensemble des cycles effectués. Les résultats représentés par les deux figures révèlent des différences significatives dans les latences de transmission entre les différents algorithmes de routage évalués. Pour notre approche de routage géographique utilisant IMOGWO, une légère variation dans les latences moyennes pendant les cycles est observée, avec une moyenne totale de 1.93 secondes. En revanche, l'approche avec MOGWO montre des latences moyennes légèrement plus basses, avec une moyenne totale

de 1.78 secondes. Les algorithmes AcNSGA-III et BFOA présentent des latences moyennes plus

FIGURE 5.25 – Comparaison des nombres d'objets actifs pendant les cycles effectués

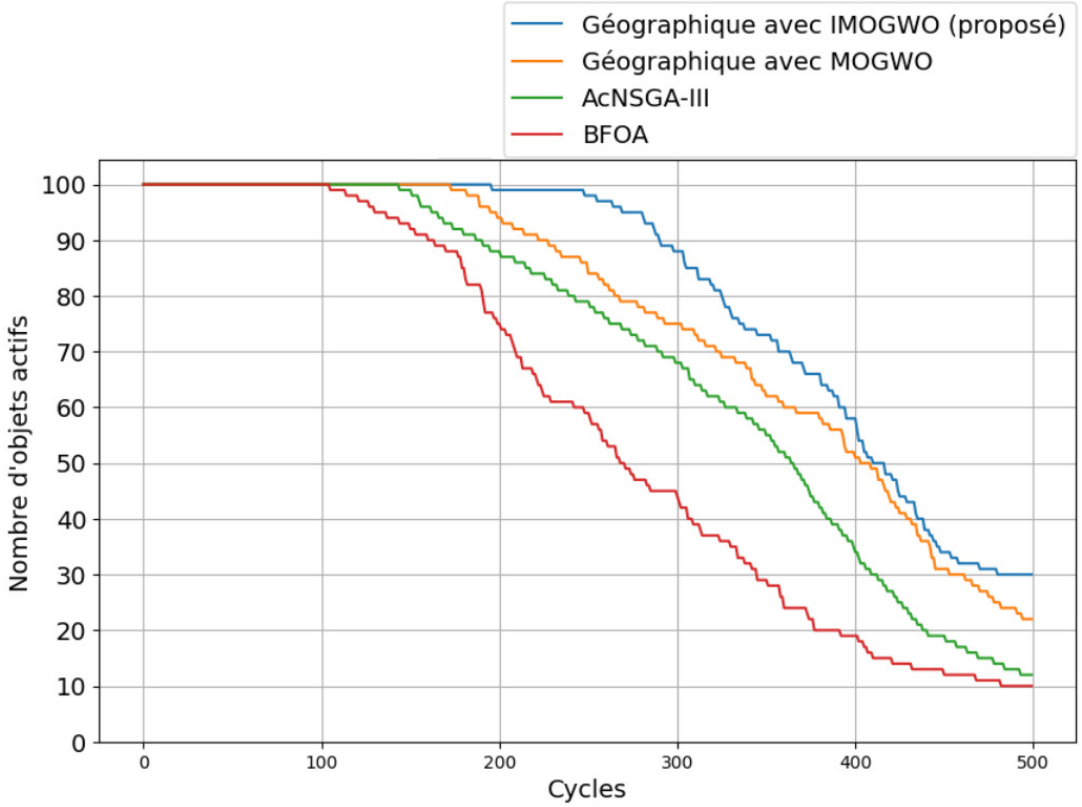
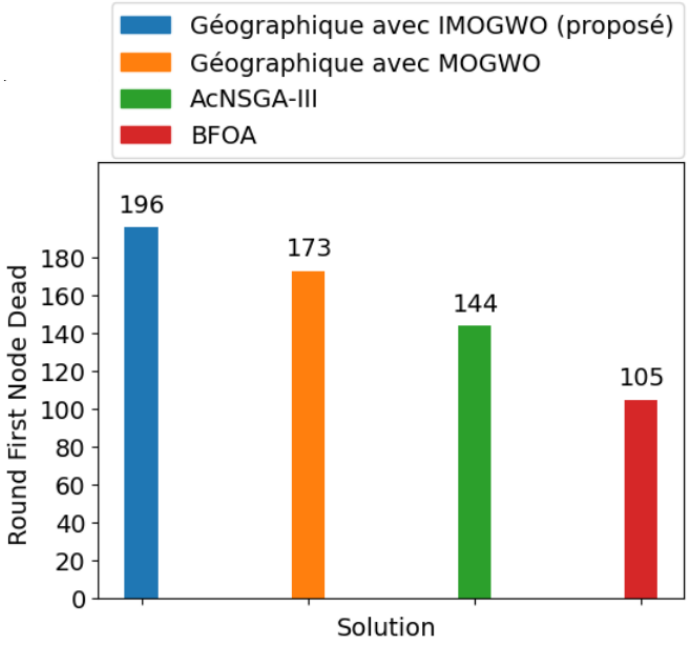


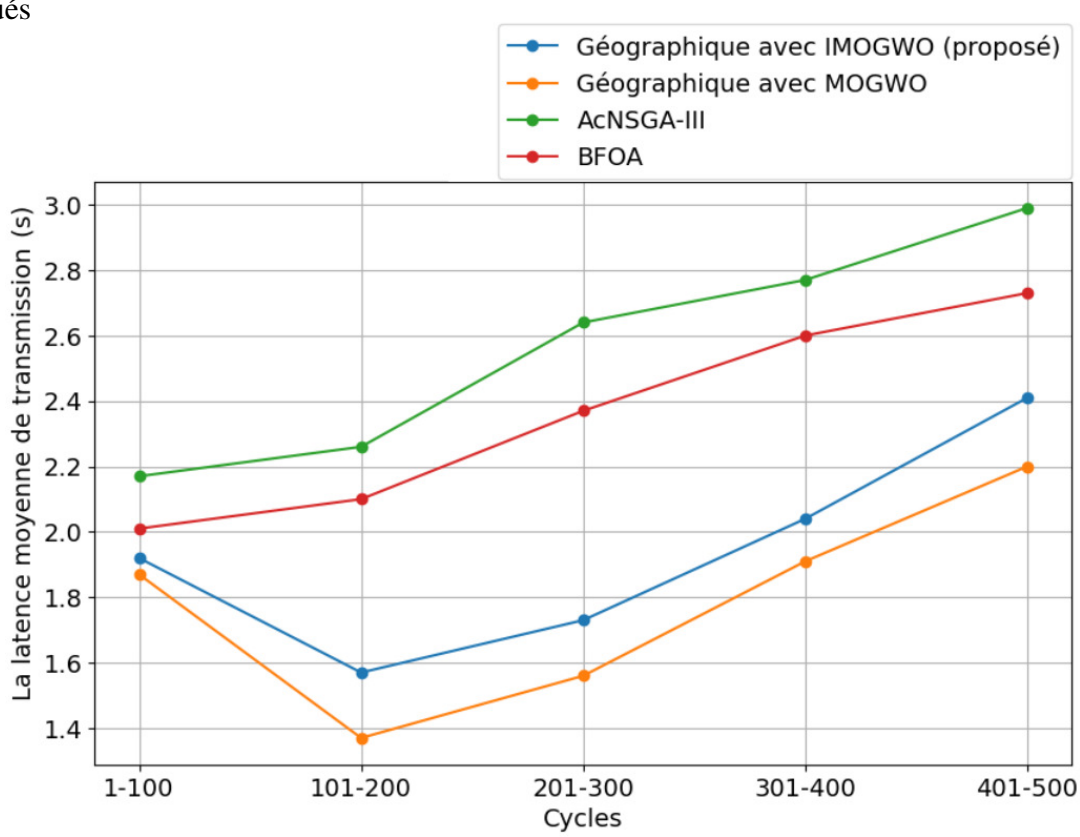
FIGURE 5.26 – Comparaison des valeurs de FND



élevées, avec des valeurs totales de 2.57 et 2.36 secondes respectivement. En conclusion, 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.

La figure 5.29 illustre une comparaison entre les algorithmes de routage évalués en termes de valeurs de PDR pendant les cycles effectués. Chaque valeur représentée dans la figure correspond au PDR évalué sur une période de 100 cycles successifs. Les valeurs de PDR représentées mettent en évidence une meilleure fiabilité fournie par notre propositon par rapport aux autres algorithmes de routage étudiés. Pour notre approche de routage géographique appliquée avec IMOGWO et avec MOGWO, le PDR est relativement élevé, avec des valeurs variantes entre 84% et 97%. En revanche, les algorithmes AcNSGA-III et BFOA présentent des PDR plus bas, variant entre 79% et 92%. Ces observations sont confirmées par les valeurs de PDR total calculées pour l'ensemble des cycles,

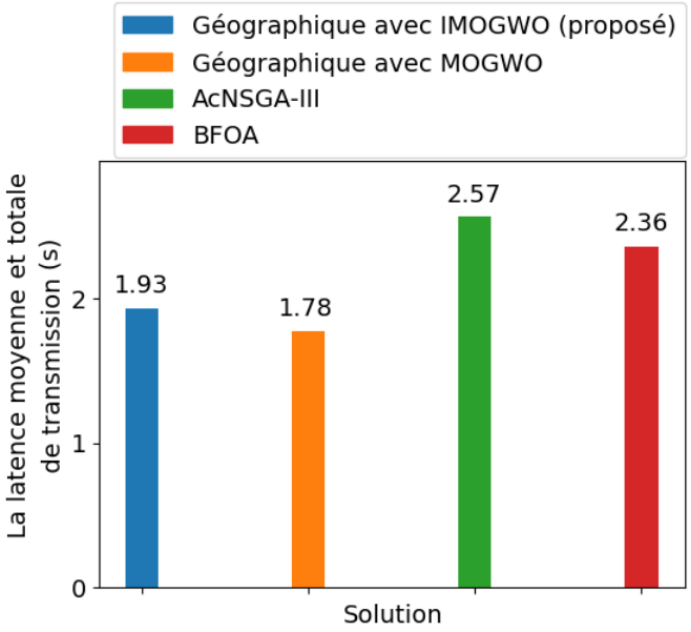
FIGURE 5.27 – Comparaison des valeurs de la latence moyenne de transmission pendant les cycles effectués



comme illustré dans la figure 5.30. 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.

La figure 5.31 montrent la variation de l'Overhead de contrôle à travers les cycles pour les différents algorithmes de routage évalués. Pour notre approche de routage géographique avec IMOGWO et MOGWO, ainsi que pour les algorithmes AcNSGA-III et BFOA, une tendance générale à la baisse de l'Overhead de contrôle au fil des cycles peut être observée. Cette diminution peut indiquer une meilleure efficacité au fur et à mesure que le système s'adapte et apprend à propos des objets du réseau. De plus, cette baisse peut également être justifiée par la diminution du nombre d'objets actifs dans le réseau, en particulier au cours des derniers cycles. En outre, en analysant les valeurs moyennes de l'Overhead de contrôle pour l'ensemble des cycles représentées dans la figure 5.32, il peut être observé que l'approche de routage géographique avec IMOGWO présente un Overhead légèrement plus élevé (21.98%) que celle avec MOGWO (21.25%). Les algorithmes AcNSGA-III

FIGURE 5.28 – Comparaison des valeurs de la latence moyenne et totale de transmission pendant les cycles effectués



et BFOA présentent des Overheads de contrôle de 20.52% et 18.59% respectivement. Cette compa-

FIGURE 5.29 – Comparaison des valeurs de PDR pendant les cycles effectués

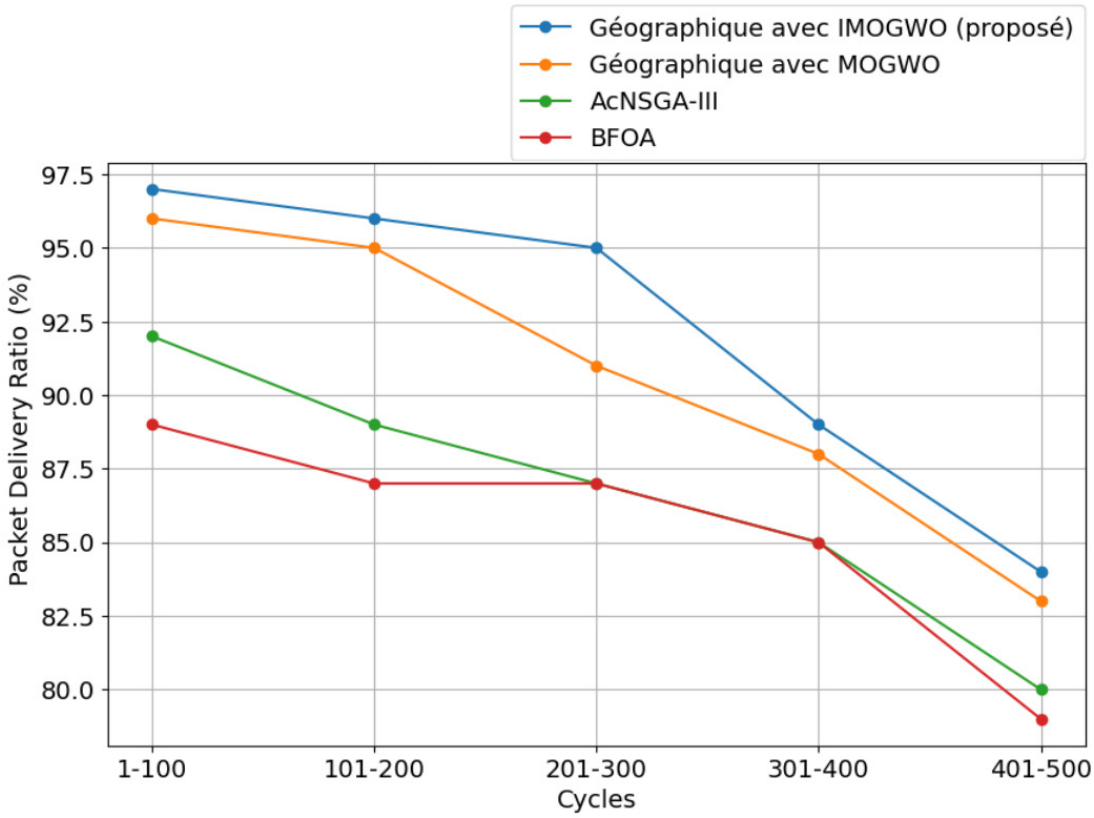
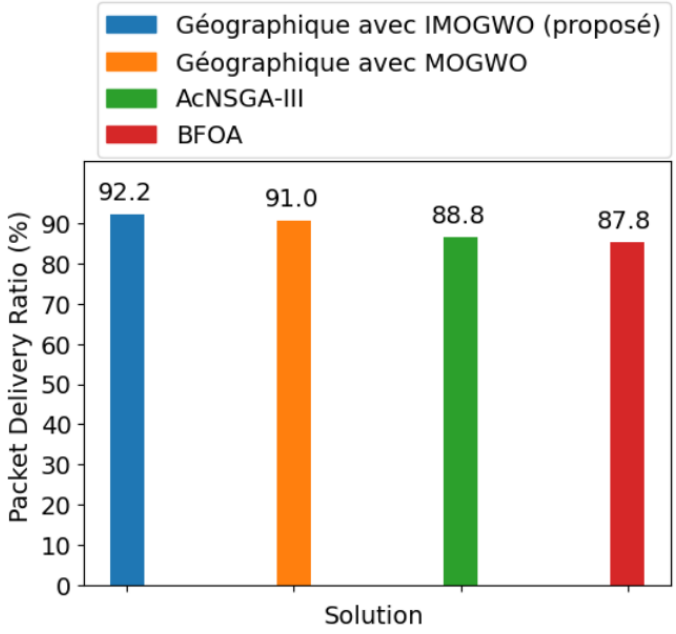


FIGURE 5.30 – Comparaison des valeurs totales de PDR

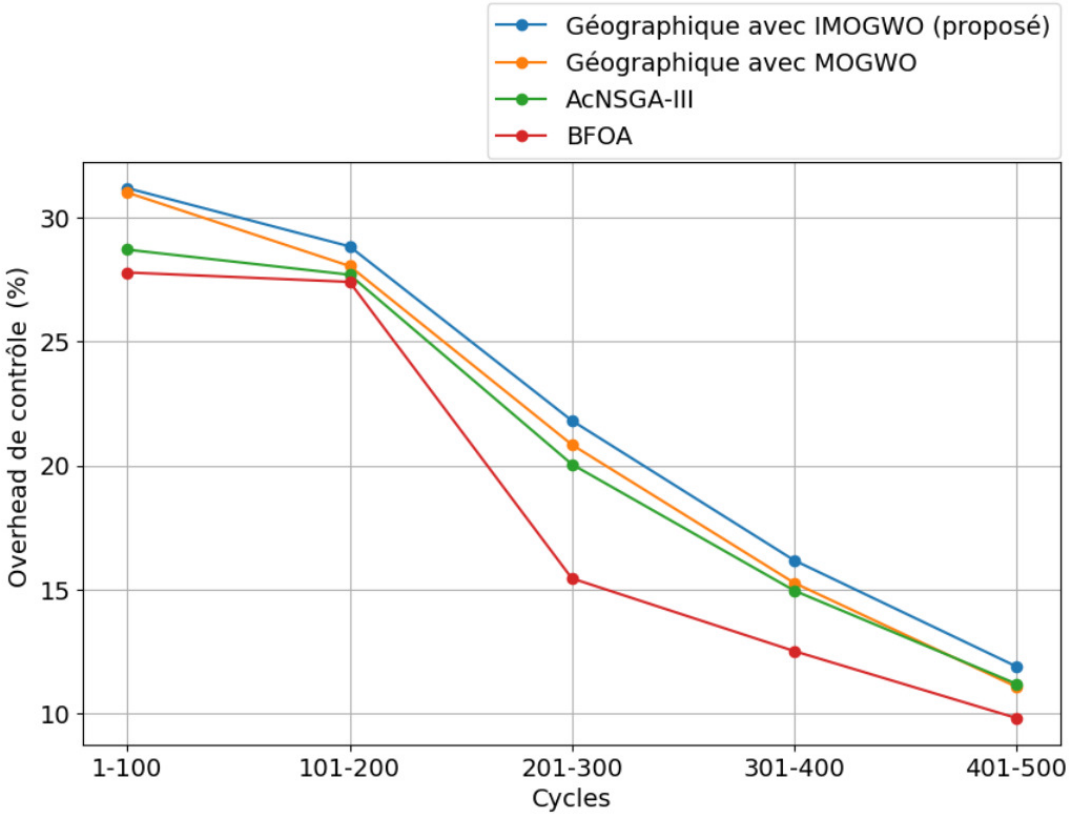


raison montre que l'algorithme BFOA est le plus efficace en termes de minimisation de l'Overhead de contrôle, suivi de près par AcNSGA-III. Les approches de routage géographique avec IMOGWO et MOGWO présentent des performances légèrement inférieures en termes d'Overhead de contrôle, mais restent compétitives par rapport aux autres algorithmes. Cela peut être justifié par le fait que notre proposition de routage géographique conserve plus d'objets actifs pour un nombre plus élevé de cycles. De plus, cela peut être expliqué par l'échange périodique des informations concernant les voisins à 1 et à 2 sauts appliquée dans le routage géographique proposé. Cependant, notre proposition reste légèrement moins performante en termes d'Overhead de contrôle par rapport aux deux autres algorithmes, car elle les surpasse en termes de fiabilité de la transmission des paquets (c'est-à-dire que la quantité de données transmises est plus élevée).

5.5.4.2 Comparaison par tests statistiques inférentiels

En effet, pour évaluer si la solution proposée représente une amélioration significative par rapport aux autres méthodes de routage, nous avons appliqué les tests statistiques inférentiels présentés dans la section 5.3.3. Nous avons appliqué des tests impliquant des comparaisons (1 x N). Le test de Friedman avec l'extension d'Iman-Davenport et la procédure post-hoc de Holm sont couram-

FIGURE 5.31 – Comparaison des valeurs de l'"Overhead de contrôle"

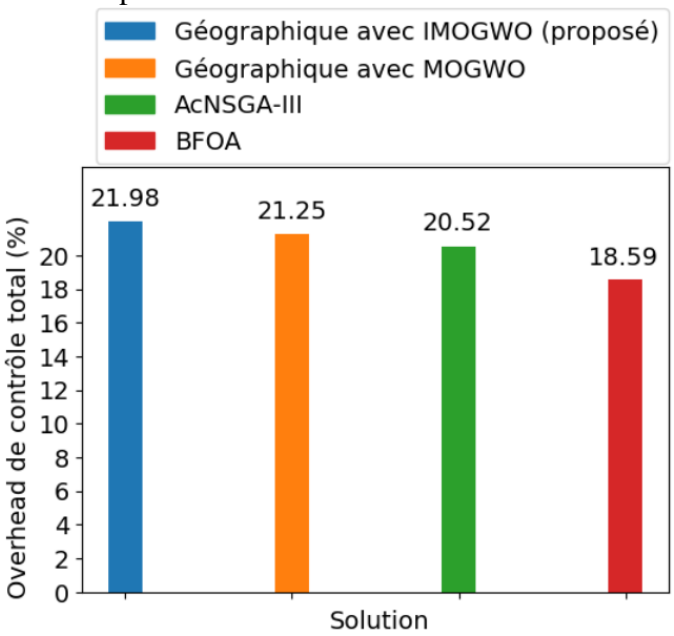


ment utilisés pour évaluer la performance relative d'un nouvel algorithme par rapport aux solutions existantes pour un problème spécifique.

a) Application du test de Friedman avec l'extension d'Iman-Davenport

Le test de Friedman avec l'extension d'Iman-Davenport offre une méthode pour comparer un algorithme à un ensemble d'autres algorithmes et pour identifier les différences statistiquement significatives entre eux. Les étapes de son application sont détaillées dans la section 5.3.3.1. Le tableau 5.23 présente les résultats statistiques du test de Friedman appliqué pour comparer notre solution de routage géographique proposée avec les autres algorithmes de routage étudiés en ce qui concerne le nombre d'objets actifs et la latence de transmission. Le rang le plus bas, indiquant une performance supérieure, est mis en gras, et le rang suivant est souligné.
Pour le critère du nombre d'objets actifs dans le réseau, nous observons que notre proposition de routage géographique avec IMOGWO obtient le rang moyen le plus bas parmi les quatre algorithmes, avec un rang de 1.42. Cela indique que, en moyenne, notre approche avec IMOGWO est classée plus haut que les autres algorithmes en termes de maintien du nombre d'objets actifs dans le réseau. L'algorithme de routage géographique proposé appliqué avec MOGWO présente le deuxième meilleur rang. En revanche, l'algorithme BFOA obtient le rang moyen le plus élevé, montrant qu'il est moins performant que les autres dans ce contexte. En ce qui concerne la latence de transmission, la proposition de routage géographique avec IMOGWO obtient un rang moyen de 1.92, ce qui est le deuxième meilleur résultat. Cependant, la proposition de routage géogra-

FIGURE 5.32 – Comparaison des valeurs totales de l'"Overhead de contrôle"



phique avec MOGWO affiche le meilleur résultat avec un rang moyen de 1.13. Les algorithmes AcNSGA-III et BFOA ont des performances inférieures avec des rangs moyens de 3.8 et 3.07 respectivement. Les statistiques de Friedman et de Iman-Davenport pour les deux critères indiquent toutes deux une différence statistiquement significative entre les algorithmes, avec une valeur de F(3.499) = 2.653, confirmant ainsi que les performances varient de manière significative entre les méthodes.
Ces analyses mettent en évidence la performance élevée de la proposition de routage géographique avec IMOGWO, bien que légèrement inférieure à celle de MOGWO en termes de latence de transmission. Les résultats montrent également que les algorithmes AcNSGA-III et BFOA présentent des performances moins satisfaisantes pour les deux critères évalués, ce qui indique la supériorité de notre proposition de routage géographique basée sur les métaheuristiques multi-objectifs.

TABLEAU 5.23 – Les rangs moyens des algorithmes de routage et le calcul des statistiques de Friedman et d'Iman-Davenport pour les métriques : le nombre d'objets actifs et la latence de transmission

Algorithme		Géographique avec IMOGWO (proposé)	Géographique avec MOGWO	AcNSGA-III	BFOA
Rang moyen	Le nombre d'objets actifs	**1.42**	2.07	2.81	3.68
	La latence de transmission	1.92	**1.13**	3.8	3.07
Statistique de Friedman pour le nombre d'objets actifs = 861.41 Statistique de Iman-Davenport pour le nombre d'objets actifs = 216.81 F(3,499) = 2.653					
Statistique de Friedman pour la latence de transmission = 1321.02 Statistique de Iman-Davenport pour la latence de transmission = 332.21 F(3,499) = 2.653					

b) Application de la procédure post-hoc de Holm

Les étapes d'application de cette procédure sont détaillées dans la section 5.3.3.2. Le tableau 5.24 représente les résultats de l'application de la procédure post-hoc de Holm au nombre d'objets actifs dans le réseau. Il est remarquable que toutes les valeurs z obtenues sont négatives, ce qui confirme que la solution proposée surpasse toutes les autres solutions de routage comparées. De plus, notre proposition de routage géographique avec IMOGWO obtient un rang moyen de 1, ce qui la place en tête, avec une valeur de p de 0.0102. Ensuite, la proposition de routage géographique avec MOGWO se classe deuxième avec un rang moyen de 2 et une valeur de p très significative ($p < 0.0001$). Les algorithmes AcNSGA-III et BFOA obtiennent respectivement les rangs moyens de 3 et 4, avec des valeurs de p très significatives ($p < 0.0001$ et $p = 1.0$). Ainsi, ces résultats confirment que la proposition de routage géographique avec IMOGWO maintient un plus grand nombre d'objets actifs dans le réseau, ce qui indique une meilleure stabilité par rapport aux autres méthodes évaluées.

En ce qui concerne le critère de la latence de transmission, le tableau 5.25 résume les résultats du test post-hoc de Holm pour la comparaison entre les algorithmes de routage évalués. Ces résultats révèlent également des différences significatives entre les algorithmes. Dans ce cas, la proposition de routage géographique avec MOGWO obtient le meilleur rang moyen de 1, avec une valeur de p de 0.0102. La proposition de routage géographique avec IMOGWO se classe deuxième avec un rang moyen de 2 et une valeur de p très significative ($p < 0.0001$). Les algorithmes AcNSGA-III et BFOA se classent respectivement troisième et quatrième, avec des valeurs de p très significatives ($p < 0.0001$ et $p = 1.0$). Ces résultats confirment que la proposition de routage géographique avec MOGWO présente la latence de transmission la plus faible parmi les méthodes évaluées, suivie de près par la proposition de routage géographique avec IMOGWO.
En conclusion, l'analyse des résultats de comparaison des algorithmes de routage met en évidence les performances différenciées des différentes méthodes évaluées. Notre proposition de routage géographique avec IMOGWO se distingue en maintenant un nombre d'objets actifs plus élevé comparé aux autres méthodes, ce qui prouve une meilleure gestion de la consommation énergétique et une meilleure stabilité. Quant à la latence de transmission, notre approche de routage géographique développée avec MOGWO présente la latence la plus faible, suivie de près par celle appliquée avec IMOGWO. Ces résultats soulignent l'efficacité de notre approche de routage géographique et hybride basée sur les algorithmes d'optimisation IMOGWO et MOGWO.

TABLEAU 5.24 – Résultats du test post-hoc de Holm pour la comparaison entre les algorithmes de routage en termes du nombre d'objets actifs

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

TABLEAU 5.25 – Résultats du test post-hoc de Holm pour la comparaison entre les algorithmes de routage en termes de latence de transmission

Algorithme	$\mathbf{z_i}$	$\mathbf{r_i}$	**Holm**	**p-value**
Géographique avec IMOGWO	0	2	0.025	< 0.0001
Géographique avec 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 Synthèse

Les résultats obtenus mettent en évidence la supériorité de notre approche de routage géographique et hybride proposée 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. 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.

5.6 Conclusion

En conclusion, ce chapitre fournit une vue d'ensemble des résultats numériques, des simulations et des expérimentations réalisées pour évaluer les approches proposées dans cette thèse. Nous avons évalué et comparé notre approche de routage réactif et multi-objectifs basée sur la métaheuristique MOGWO avec NSGA-III. Les résultats ont démontré la supériorité de notre proposition par rapport à NSGA-III. Ensuite, l'efficacité de notre algorithme proposé, IMOGWO, dans la résolution des problèmes de référence DTLZ a été démontrée à travers une série de tests. De plus, nous avons analysé les performances de notre approche de localisation en intérieur, qui combine le DL et les métaheuristiques, ainsi que les résultats de l'algorithme de routage géographique hybride et multi-objectifs proposé. Les résultats ont confirmé la supériorité de nos solutions proposées par rapport aux autres solutions comparées, mettant ainsi en évidence les points forts et les points faibles des différentes approches évaluées. Ces résultats seront synthétisés dans la conclusion générale, prochain chapitre (section 6.1). Les discussions et les analyses présentées contribuent à une meilleure compréhension des performances des solutions proposées et ouvrent la voie à de futures recherches dans ce domaine passionnant.

Chapitre 6

Conclusions générales et orientations pour les recherches futures

Sommaire

Dans cette thèse, nous avons abordé et étudié les défis relatifs au routage et à la localisation en milieu intérieur dans les réseaux IoT, deux aspects cruciaux pour l'efficacité et la fiabilité de ces réseaux en constante évolution. Nous avons exploré l'optimisation métaheuristique et les techniques de DL pour faire face à ces défis. Pour ce faire, la thèse s'appuie sur une revue approfondie de la littérature pour situer ces problématiques dans leur contexte et propose ensuite des contributions relatives à ce sujet. La première contribution implique une modélisation mathématique du problème de routage dans les réseaux IoT, offrant un cadre structuré pour la conception et l'optimisation des algorithmes de routage. Ensuite, une approche de routage multi-sauts, réactif et multi-objectifs a été développée en utilisant l'algorithme MOGWO [2]. Par la suite, une amélioration de cet algorithme, nommée IMOGWO, a été proposée pour résoudre des problèmes comportant un plus grand nombre d'objectifs. Une méthode a également été proposée pour optimiser l'entraînement des modèles de DL à l'aide d'une métaheuristique. Cette méthode sert de base à une approche de localisation développée en combinant l'optimisation métaheuristique avec le DL pour prédire avec précision les positions des objets dans un réseau IoT. Enfin, une approche de routage géographique hybride et multi-objectifs a été proposée, exploitant notre proposition de localisation précise des objets pour optimiser le processus de routage en tenant compte de divers objectifs simultanément. Ces approches ont été évaluées à travers des expériences réelles et des simulations, démontrant leurs performances comparées à d'autres solutions existantes. Les contributions et les résultats des évaluations sont résumés dans la section suivante.

6.1 Conclusions et résultats

Les contributions principales et les résultats correspondants de notre travail de recherche se résument comme suit :

— Nous avons mené une revue approfondie de la littérature, débutant par une introduction générale à l'IoT. Ensuite, nous avons étudié en détail le routage et la localisation en intérieur dans les réseaux IoT. Par la suite, nous avons abordé les aspects liés à l'optimisation et au DL. L'état de l'art a étudié également les travaux récents qui sont liés aux thèmes abordés.

— Une modélisation mathématique du routage dans les réseaux IoT a été proposée, identifiant et formalisant les caractéristiques du réseau IoT, ainsi que les objectifs clés, les variables de décision et les contraintes associées. Cette modélisation a servi à rendre les approches proposées plus structurées et à simplifier leur évaluation. Elle met l'accent sur divers objectifs : la progression 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.

— Une approche de routage a été présentée, appliquant l'algorithme MOGWO [2] pour traiter le routage comme un problème d'optimisation multi-objectifs. L'objectif a été de développer un algorithme de routage multi-sauts, réactif et multi-objectifs dans un réseau IoT à topologie hybride. Les performances de cette approche de routage modélisée avec MOGWO ont été évaluées à travers des expérimentations réelles et comparées à celles de NSGA-III [59]. Les résultats obtenus mettent en évidence que MOGWO offre une solution plus efficace et prometteuse pour le routage dans les réseaux IoT par rapport à NSGA-III. Cela se traduit par une meilleure efficacité énergétique, un équilibrage de charge amélioré, des délais de transmission réduits, des communications plus fiables et stables, ainsi qu'une répartition plus efficace des objets dans le réseau.

— Nous avons développé une amélioration de la métaheuristique multi-objectifs MOGWO, nommée IMOGWO. Il implique des ajustements des équations fondamentales de MOGWO. L'objectif principal de cette amélioration est d'accroître l'efficacité de MOGWO dans la résolution de problèmes complexes à multiples objectifs, assurant ainsi une convergence efficace et une distribution optimale des solutions obtenues. 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]. Nous avons mené une série de tests sur ces problèmes en variant le nombre d'objectifs et le problème étudié. Les résultats obtenus avec IMOGWO ont été comparés avec ceux obtenus par les optimiseurs MOGWO, DMOPSO [63], MOEA/D-IEp [69], MOEA/DD [66] et ESPEA [64] en termes des indicateurs IGD et NHV. Des com-

paraisons standards et des tests statistiques inférentiels ont été effectués pour montrer que IMOGWO présente une amélioration notable par rapport aux autres optimiseurs en garantissant une convergence efficace et une distribution optimale des solutions obtenues.

— Nous avons proposé une méthode d'optimisation de l'entraînement des modèles de DL en appliquant une métaheuristique mono-objectif. 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. Cette méthode a été utilisée ensuite pour développer l'approche de localisation en intérieur dans les réseaux IoT.

— Nous avons proposé une approche de localisation en intérieur pour les réseaux IoT. Elle est construite sur notre précédente approche, puisqu'elle entraîne un modèle DL avec des mesures de temps de vol UWB en appliquant l'algorithme d'optimisation métaheuristique GWO [3]. Nous avons évalué les performances de notre solution en la comparant à trois autres méthodes de localisation largement utilisées : une méthode basée sur le DL [154], le IWCL [155], et une méthode de multilatération [156]. Pour démontrer la supériorité de notre proposition, nous avons réalisé des comparaisons standards et des tests statistiques inférentiels. 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. En effet, notre méthode présente une erreur moyenne de seulement 0.057 mètres entre les positions réelles et prédites, avec une précision de localisation de 98.92%. Ces résultats dépassent ceux des solutions de localisation en intérieur récentes, dont les meilleures performances présentent une erreur moyenne de quelques dizaines de centimètres.

— Nous avons développé une approche de routage géographique hybride et multi-objectifs conçue pour les réseaux IoT à architecture maillée. La nature géographique de l'algorithme proposé nécessite la connaissance des positions des objets dans le réseau, ce qui est assuré par notre approche de localisation en intérieur proposée. Le processus de sélection des routes multi-sauts est optimisé selon divers objectifs simultanément à l'aide de l'algorithme proposé IMOGWO. Des simulations hybrides, combinant des éléments simulés et réels, ont été réalisées pour évaluer l'approche de routage géographique et hybride 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 BFO [109]. Cette comparaison s'est basée sur divers indicateurs d'évaluation tels que l'efficacité, la fiabilité, l'optimisation de l'énergie, l'équilibrage de charge et la stabilité. Les comparaisons standards et les analyses par les tests statistiques inférentiels ont montré que IMOGWO

surpasse MOGWO dans la résolution du problème de routage géographique multi-objectifs et ont confirmé la supériorité de notre approche de routage géographique et hybride par rapport aux autres algorithmes comparés. Notre proposition a démontré son efficacité en termes de gestion de la consommation énergétique, de répartition de la charge, de latence de transmission et de taux de livraison des paquets. En dépit d'une légère augmentation des données de contrôle, les approches de routage géographique, utilisant MOGWO et IMOGWO, offrent une meilleure fiabilité de transmission, assurant ainsi une meilleure qualité de service dans l'ensemble du réseau.

En résumé, les évaluations de nos approches proposées par des expérimentations réelles et des simulations hybrides, l'utilisation d'indicateurs de performance adaptés, ainsi que les comparaisons avec des solutions existantes et les analyses statistiques effectuées, mettent en évidence le bon comportement et l'efficacité de nos approches sur les défis de routage et de localisation en intérieur.

6.2 Orientations pour les recherches 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, 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 DL 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 DL 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 DL, afin d'améliorer la précision de la localisation en milieu intérieur dans les réseaux IoT.

— Routage basé sur le contexte : explorer des méthodes de routage qui prennent en compte le contexte spécifique de l'environnement intérieur, comme la disposition des obstacles, la densité du trafic et la qualité des liens, pour optimiser la sélection des chemins de communication.

— 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.

Publications (dans cette thèse)

Revues internationales

— 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), [Soumission en cours d'évaluation, soumise le 12 novembre 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) [Soumission en cours d'évaluation, soumise le 27 Juin 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) [Soumission en cours d'évaluation, soumise le 27 Août 2024]

Conférences Internationales

— 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.

Conférences Nationales

— 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.

Bibliographie

[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 Commu-*

nications. 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 multiobjective 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.

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Université de Toulouse

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.