DEMOCRATIC AND POPULAR REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND RESEARCH

UNIVERSITY OF SAAD DAHLAB BLIDA - 1
Faculty of Sciences
Department of Computer Science
LRDSI

DOCTORAL DISSERTATION (LMD)

Specialization : Networks and Distributed Systems

MULTI-OBJECTIVE OPTIMIZATION OF THE DEPLOYMENT OF SENSOR NETWORKS DEDICATED TO THE SURVEILLANCE OF SENSITIVE SITES.

Presented by :
Slimane Charafeddine Benghelima

Examination Committee :

Hafida Abed	Professor, University of Blida 1	President
Narhimene Boustia	Professor, University of Blida 1	Examiner
Bouabdellah Kechar	Professor, University of Oran 1	Examiner
Mohamed Ould Khaoua	Professor, University of Blida 1	Supervisor
Ali Benzerbadj	Associate Professor, University of Ain Temouchent	Co-Supervisor

Algeria, Blida, 2024

I dedicate this work a testimony to my dearest father BENGHELIMA HABIB and mother ZERIKAT MAMA for all of their sacrifices, love, tenderness, support, and prayers throughout my education. May Allah grant you long life and good health.

To my brother BENGHELIMA MOHAMED AMINE, who offered guidance, encouragement and motivation to complete this doctoral study.

Acknowledgements

I am profoundly grateful to my supervisory team for their unwavering support and guidance throughout my doctoral journey. I am extremely lucky to have had the opportunity to work under their supervision. Firstly, I extend my sincere appreciation to my main supervisor, Prof. OULD KHAOUA Mohamed, from Saad Dahlab University- Blida 1, Algeria, whose invaluable advice, availability, and trust in my abilities have been instrumental in shaping my research. Prof. OULD KHAOUA's extensive research experience and expertise have not only enriched my understanding of the research methodology but have also fostered in me the resilience necessary for the effective formulation, articulation, and defense of research ideas.

I also extend my heartfelt thanks to my first co-supervisor, Dr. BENZERBADJ Ali, from Ain Temouchent Belhadj Bouchaib University, Algeria, for his invaluable guidance and support throughout my doctoral research. Dr. BENZERBADJ's mentorship has played a pivotal role in shaping the direction of my work over the past five years. His insightful feedback and encouragement have been indispensable to my academic growth, and I am deeply grateful for his technical contributions during this research.

Additionally, I express my sincere appreciation to my second co-supervisor, Prof. BAALA Oumaya, from Université de Technologie de Belfort-Montbéliard, France, for her mentorship, support, and guidance during my doctoral studies. I am grateful for her invaluable contributions to my doctoral journey.

I extend my sincere thanks to Professor Hafida Abed, from the University of Blida 1, for accepting to be the Chair of my examination committee. My heartfelt gratitude goes to Professor Narhimene Boustia, from the University of Blida 1, and Professor Bouabdellah Kechar, from the University of Oran 1, for agreeing to be members of the examination committee and for the time they dedicated to examining this thesis.

I would also like to thank all the members of LRDSI and the Department of Computer Science at Saad Dahlab University- Blida 1 for their great support, and everyone who has contributed directly or indirectly to the completion of this research thesis. Your support, whether through insightful discussions, constructive feedback, or moral encouragement, has been instrumental in shaping this work, and I am truly thankful for your involvement.

Abstract

This research endeavors to enhance the efficiency of Wireless Sensor Networks (WSNs) deployment specifically tailored for Smart Car Parks (SCP) surveillance applications. Traditional deployment methods often follow a sequential approach, leading to suboptimal solutions in terms of node placement, network coverage, and connectivity. To address these limitations, our proposed approach advocates the simultaneous placement of Sensor Nodes (SNs) and Relay Nodes (RNs) to minimize the overall number of required nodes while ensuring sufficient coverage and connectivity throughout the car park area.

The core of our methodology lies in the development of a novel optimization framework that integrates both deterministic and meta-heuristic techniques. Firstly, we introduce a Multi-Objective Linear Programming model (MOLP) capable of efficiently solving smaller instances of the deployment problem, optimizing objectives such as the number of SNs, number of RNs, and network diameter simultaneously. To tackle larger instances of the deployment problem, we complement MOLP with the Greedy Chaos Whale Optimization meta-heuristic (GCWOA), which is designed to handle complex optimization scenarios effectively.

GCWOA combines several optimization strategies to achieve superior deployment solutions. Initially, a Greedy algorithm is employed for the initial placement of nodes, facilitating a quick but reasonably good solution. Subsequently, a Chaos Local Search (CLS) technique is integrated, leveraging chaos maps to explore the solution space effectively and refine the initial placement. Furthermore, the meta-heuristic divides the solution population into two sub-populations, each subjected to different optimization techniques. One sub-population undergoes CLS for further refinement, while the other utilizes the Whale Optimization Algorithm (WOA) to exploit the search space more intensively, thereby enhancing the solution quality.

An extensive comparative performance evaluation against established meta-heuristics such as the basic WOA, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO) demonstrate the superiority of GCWOA. On average, GCWOA achieves a significant fitness improvement of 23.45%, 28.75%, and 26.32% compared to basic implementations of WOA, GA, and PSO, respectively. Moreover, GCWOA exhibits faster convergence rates and reduced runtimes, highlighting its effectiveness in tackling SCP surveillance deployment challenges efficiently and reliably.

Keywords : Internet of things, Smart car park surveillance, Wireless sensor networks, Simultaneous deployment, Linear programming, Chaos local search, Whale optimization algorithm.

Résumé

Cette recherche s'efforce d'améliorer l'efficacité du déploiement des réseaux de capteurs sans fil spécifiquement adaptés aux applications de surveillance des parkings intelligents. Les méthodes de déploiement traditionnelles suivent souvent une approche séquentielle, conduisant à des solutions sous-optimales en termes de placement des nœuds, de couverture des cibles et de connectivité. Pour remédier à ces limitations, notre approche proposée préconise le placement simultané de nœuds capteurs et de nœuds relais pour minimiser le nombre total de nœuds requis tout en assurant une couverture et une connectivité suffisantes dans toute la zone a surveillé.

Le cœur de notre méthodologie réside dans le développement d'un nouveau framwork d'optimisation qui intègre à la fois des techniques déterministes et méta-heuristiques. Tout d'abord, nous introduisons un modèle de programmation linéaire multi-objectifs capable de résoudre efficacement de plus petites instances du problème de déploiement, optimisant des objectifs tels que le nombre nœuds capteurs, le nombre de nœuds relais et le diamètre du réseau simultanément. Pour aborder des instances plus grande du problème de déploiement, nous complétons avec une méta-heuristique d'optimisation de la baleine combiné avec un algorithme greedy, une recherche locale et la théorie du chaos, conçue pour gérer efficacement des scénarios d'optimisation complexes.

Notre proposition combine plusieurs stratégies d'optimisation pour atteindre des solutions de déploiement supérieures. Initialement, un algorithme greedy est utilisé pour le placement initial des nœuds, facilitant l'obtention d'une population de solutions rapidement mais raisonnablement bonne. Ensuite, une technique de recherche locale chaotique est intégrée, exploitant la théorie du chaos pour explorer efficacement l'espace des solutions et affiner le placement initial. De plus, notre méta-heuristique divise la population de solutions en deux sous-populations, chacune soumise à des techniques d'optimisation différentes. Une sous-population subit hybridation entre une recherche locale et la théorie du chaos pour un affinement supplémentaire, tandis que l'autre utilise l'algorithme d'optimisation de la baleine (WOA) pour exploiter plus intensément l'espace de recherche, améliorant ainsi la qualité de la solution.

Une évaluation de performance comparative approfondie par rapport aux méta-heuristiques établies telles que WOA, l'algorithme génétique (GA) et l'optimisation par essaim de particules (PSO) démontre la supériorité de notre méta-heuristique. En moyenne, notre méta-heuristique réalise une amélioration significative de la qualité de solutions de 23,45%, 28,75% et 26,32% par rapport aux implémentations de base de WOA, GA et PSO, respectivement. De plus, notre métaheuristique présente des taux de convergence plus rapides et un temps d'exécution réduit, mettant en évidence son efficacité dans la résolution efficace et fiable des défis de déploiement de surveillance SCP.

Mots-clés :Internet des objets, surveillance de parking intelligent, réseaux de capteurs sans fil, déploiement simultané, programmation linéaire, recherche locale, algorithme d'optimisation de la baleine.

ملخص

تسعى هذه الأطروحة إلى تعزيز كفاءة نشر شبكات الاستشعار اللاسلكية مصممة خصيصًا لتطبيقات مراقبة مواقف السيارات الذكية . تتبع الطرق التقليدية للنشر نهجًا تسلسليًا، مما يؤدي إلى حلول غير مثلى من حيث وضع أجهزة الإستشعار و أجهزة نقل البيانات، وتغطية الأهداف والتوصيل. للتغلب على هذه القيود، تدعو الطريقة المقترحة لدينا إلى وضع متزامن لأجهزة الإستشعار و أجهزة نقل البيانات لتقليل عددها الإجمالي مع ضمان تغطية الأهداف وتوصيل كافٍ لأجزاء الشبكة في جميع أنحاء منطقة موقف السيارات.

يكمن جوهر منهجيتنا في تطوير إطار تحسين جديد يدمج تقنيات محددة وذاتية الهيمنة. أولاً، نقدم نموذج برمجة خطية متعددة الأهداف قادر على إيجاد حلول دقيقة لحالات صغيرة لمشكلة النشر، محسناً أهداف مثل عدد أجهزة الإستشعار، و أجهزة نقل البيانات، وقطر الشبكة في وقت واحد. للتعامل مع حالات أكبر لمشكلة النشر،لجأنا لتصميم طريقة تقريبية تعتمد على خوارزميات جشعة، البحث المحلي و نظرية فوضى الكون، والتي تم تصميمها للتعامل مع سيناريوهات المعقدة بفعالية.

تجمع طريقتنا التقريبية بين عدة استراتيجيات تحسين لتحقيق حلول نشر متفوقة. في البداية، يُستخدم خوارزميات جشعة للوضع الأولي لأجهزة الإستشعار و أجهزة نقل البيانات، مما يسهل الحصول على مجموعة من الحلول أولية بشكل سريع ولكن مقبول بشكل معقول. بعد ذلك، يتم دمج تقنية البحث المحلي و نظرية فوضى الكون، مستفيدين من خرائط الفوضى لاستكشاف مساحة الحلول بفعالية . تقسم مجموعة الحلول الأولية إلى مجموعتين فرعيتين، تخضع كل منها لتقنيات تحسين مختلفة. تخضع مجموعة فرعية لتقنية البحث المحلي و نظرية فوضى الكون، بينما تستخدم الأخرى خوارزمية الحوت (WOA) لاستغلال مساحة البحث بشكل أكثر تكثيفًا، مما يعزز جودة الحل.

تشير التقييمات الأدائية المقارنة الوافرة مقابل التقنيات الذاتية الحالية مثل WOA و GA و PSO إلى تفوق طريقتنا . بشكل متوسط، تحقق طريقتنا تحسينًا كبيرًا في اللياقة بنسبة 23.45 %، 28.75 % و 26.32 % مقارنة لـ WOA و GA و PSO على التوالي. علاوة على ذلك، تظهر طريقتنا معدلات تقارب أسرع وزمن تشغيل أقل، مما يسلط الضوء على فعاليته في مواجهة تحديات نشر مراقبة بكفاءة وموثوقية.

الكلمات المفتاحية

أنترنت الأشياء، مراقبة مواقف السيارات الذكية، شبكات الاستشعار اللاسلكية، النشر متزامن، البرمجة الخطية ، البحث المحلي ، خوارزمية الحوت.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapitre 1

General Introduction

Wireless Sensor Networks (WSNs) play a vital role in enabling various emerging applications such as smart homes and cities [1], healthcare monitoring [2], disaster management systems, and smart car park [3]. Among the components of WSNs, Sensor Nodes (SNs) hold particular significance, serving as the primary entities responsible for data collection whereas Relay Nodes (RNs) are responsible for routing collected data to the sink node for further processing and analysis. The deployment of WSNs for monitoring sensitive sites has emerged as a crucial area of research investigation, necessitating a comprehensive exploration of multi-objective optimization techniques [4]. The motivations for the present research work is outlined below.

1.1 Motivations

The advent of smart car parks marks a transformative leap in urban infrastructure management, promising various benefits ranging from optimized parking space utilization to reduced traffic congestion and environmental impact. However, amid the excitement surrounding these advancements, it is essential to acknowledge the inherent vulnerabilities and potential risks associated with such complex systems, particularly concerning safety and security. One of the most pressing concerns in this regard is the threat of fire incidents within parking facilities, which can escalate rapidly, posing grave dangers to property, vehicles, and, most importantly, human lives.

The integration of WSNs tailored for surveillance in car parks has emerged as a critical solution to mitigate these risks effectively. By strategically deploying SNs equipped with advanced fire detection capabilities throughout the smart car park infrastructure, operators can establish a comprehensive surveillance system capable of promptly identifying and alerting authorities to any signs of fire or smoke. These SNs can be programmed to detect subtle changes in temperature, air quality, and other relevant parameters, enabling them to provide early warnings of potential fire outbreaks before they escalate into full-blown emergencies. Moreover, the deployment of WSNs for fire surveillance in smart car parks serves as a testament to our collective responsibility to prioritize safety and well-being in urban environments.

On the other hand, to guarantee a high degree of surveillance with minimal cost, the deployment of WSNs must be optimized in terms of cost and performance. Multi-objective optimization provides the approriate framework for effectively handling competing goals arising in such deployement scenarios . Achieving an appropriate deployment strategy involves minimizing the number of SNs, and RNs, to manage costs and network diameter to minimize communication latency. By addressing these goals simultaneously, one can hope to achieve a compromise between increasing network coverage, reducing resource use/cost, and guaranteeing effective data transfer.

1.2 Problem Statement

This thesis endeavors to delve into the complexities surrounding the optimization of WSNs deployment, specifically focusing on the strategic placement of SNs and RNs for monitoring sites such as smart car parks. Historically, research in this field has predominantly followed a sequential placement technique, wherein SNs are initially positioned to cover designated target points, followed by the placement of RNs to maintain connectivity between deployed SNs and the central sink node. However, this traditional approach has been found to suffer from a serious limitation that restrict its effectiveness. Notably, the sequential optimization of SNs and RNs may lead to suboptimal solutions, as the placement of SNs does not always account for the subsequent positioning of RNs. Consequently, SNs may end up being situated in locations that do not contribute optimally to network topology, while optimizing RNs may necessitate additional nodes, thereby escalating deployment costs and expanding the network diameter.

Motivated by the above observations, this research work advocates the adoption of simultaneous placement techniques of SNs and RNs. By considering the simultaneous placement of both types of nodes, this alternative approach seeks to address the inherent dependencies between SNs and RNs, thereby optimizing the overall network topology. The primary objective is to simultaneously identify optimal positions for SNs and RNs that ensure complete coverage of target points while facilitating efficient connectivity to the sink node. Crucially, our proposed novel technique aims to achieve these objectives while minimizing deployment costs and network diameter.

Indeed, addressing the optimization of WSNs deployment presents a combinatorial challenge, as it involves determining the most efficient placement of SNs and RNs within a given area. Given the vast number of possible configurations, finding optimal solutions becomes increasingly complex. Consequently, it is imperative to develop efficient algorithms that are capable of exploring these configurations and identifying optimal deployment strategies within a reasonable timeframe.

1.3 Contributions

In this research work, a novel technique is adopted to address the complex problem of network deployment optimization in WSNs. One key contribution made by this study is the development of a fitness function capable of simultaneously addressing multiple objectives, particularly focused on minimizing the number of SNs, RNs, and network diameter. This fitness function is designed to optimize the deployment strategy while considering coverage and connectivity constraints.

To further enhance the optimization process, a multi-objective linear programming model is formulated, specifically tailored to handle small-scale problem instances, such as those within a 10x10 grid area. Additionally, the model incorporates the crucial constraint of K-coverage to ensure redundancy in SNs covering the same target point, thereby improving system reliability and robustness.

To efficiently tackle the large-scale problem instances, a novel greedy algorithm is proposed, which iteratively places nodes while satisfying coverage and connectivity constraints. This new algorithm serves as the basis for generating an initial population, which is subsequently divided into two sub-populations. The first sub-population undergoes a local search using the Chaos Theory to enhance the solution quality, while the second sub-population benefits from the advantages of the Whale Optimization Algorithm (WOA), such as exploration and exploitation, to further refine the solution space.

Moreover, the time complexity for calculating the objective function and ensuring compliance with the imposed constraints is computed. The analysis of the performance results provides valuable insights into the scalability and efficiency of the proposed approach, demonstrating its feasibility for practical deployment scenarios.

1.4 Thesis Outline

This dissertation is organized into seven chapters, each providing a comprehensive exploration of distinct yet interrelated topics. Chapter 2 discusses an overview of WSNs, delving into the internal structure of SNs, RNs, and the sink node. The routing in single-tiered and two-tiered WSNs is then described. After that, the chapter discusses communication models, routing, and the different types of deployment of WSNs.

Chapter 3 provides an overview of combinatorial optimization techniques. This chapter presents the theoretical foundations of optimization problems and metaheuristics. The chapter starts with an introduction to optimization problems, algorithmic complexity, and various combinatorial optimization problems. Related topics include multiobjective optimization problems, exact resolution methods, heuristics, and metaheuristics, encompassing both single-solution and population-based approaches. The chapter concludes with an exploration of hybrid metaheuristics, offering a comprehensive understanding of optimization techniques.

Chapter 4 deals with the state-of-the-art on the deployment of WSNs for surveillance applications. Focusing on the deployment of WSNs for practical surveillance applications, this chapter conducts a thorough review of the current suggested approaches to tackle the deployment problem. It explores different approaches for WSNs deployment, such as linear programming, and the different metaheuristic methods. The chapter further examines placement optimization techniques, considering both SNs and RNs.

Chapter 5 deals with simultaneous SNs and RNs deployment optimization using linear programming. Focusing on a practical case study, this chapter introduces a modeling technique using a linear programming approach for the simultaneous deployment of SNs and RNs. The chapter provides insights into the experimental set up, presents performance results, and conducts an analysis of the collected results.

Chapter 6 deals with the simultaneous SNs and RNs deployment optimization using a hybrid metaheuristic. It details the system model, sensing model, and communication model. This chapter formulates the optimization problem and proposes a solution using a hybrid metaheuristic approach. The Whale Optimization Algorithm (WOA) with the proposed enhanced version using Chaos Theory is employed, and the chapter presents experimental results and conducts an analysis to interpret the results.

The final Chapter 7 summarizes the key findings gleaned from the present research study, based on the exploration of combinatorial optimization of WSNs deployment for surveillance applications. After that, the chapter outlines potential future research directions, ensuring the work contributes to the ongoing advancement of knowledge in the field.

Publications (from this research)

Scholarly Journal

S.C. Benghelima, M. Ould Khaoua, A. Benzerbadj, and O. Balaa, Simultaneous sensor and relay nodes deployment for Smart Car Park surveillance. Evolutionary Intelligence, Springer (2023), doi : 10.1007/s12065-023-00853-z.

International Conferences

S.C. Benghelima, M. Ould Khaoua, A. Benzerbadj, O. Balaa, and J. Ben Othman, Optimization of the Deployment of Wireless Sensor Networks Dedicated to Fire Detection in Smart Car Parks using Chaos Whale Optimization Algorithm, Proc. IEEE International Conference on Communications (ICC'2022), Seoul, Korea, Republic of, 2022, pp. 3592-3597, doi :10.1109/ ICC45855.2022.9838744.

S.C. Benghelima, M. Ould Khaoua, A. Benzerbadj, and O. Balaa, Multi-objective Optimisation of Wireless Sensor Networks Deployment : Application to fire surveillance in smart car parks, Proc. 2021 International Wireless Communications and Mobile Computing (IWCMC), IEEE Computer Society, Harbin City, China, 2021, pp. 98-104, doi : 10.1109/ IWCMC51323.2021. 9498747.

S.C. Benghelima, M. Ould Khaoua, A. Benzerbadj, and O. Balaa, Multi-Objective Optimization of the Deployment of Wireless Sensor Networks for Fire Surveillance in Smart Car Parks. ROADEF - 22nd annual congress of the French Society for Operational Research and Decision Support, Apr 2021, Mulhouse (online), France.

S.C. Benghelima, M. Ould Khaoua, A. Benzerbadj, and O. Balaa, Minimal Node Deployment in Wireless Sensor Networks Under Coverage and Connectivity Constraints. ROADEF-23nd annual congress of the French Society for Operational Research and Decision Support, Feb. 2022, Lyon, France.

Chapitre 2

Overview of Wireless Sensor Networks

2.1 Introduction

Wireless Sensor Networks (WSNs) are essential enablers for emerging applications such as smart homes and cities [1], healthcare monitoring [2], disaster management systems [3], and smart car park management [5]. The senors nodes is most important entity of WSNs. Deployment of WSNs for sensitive site monitoring is an important field of study that requires a thorough investigation of multi-objective optimization techniques [4].

In the current landscape of heightened security concerns, effective monitoring of sensitive locations requires not just the judicious allocation of Sensor Node (SN) resources but also a meticulous assessment of diverse and, at times, conflicting objectives. This research meticulously explores the intricacies of multi-objective optimization in the context of deploying SN networks [1].

Sensitive sites, ranging from military installations to critical infrastructure, require vigilant surveillance to safeguard against potential threats. Traditional approaches to deploy SN networks often fall short in addressing the diverse and dynamic challenges posed by these environments. Multi-objective optimization emerges as a promising avenue, offering the prospect of simultaneously optimizing conflicting objectives such as coverage, energy efficiency, and cost-effectiveness [6].

The primary objective of this research is to formulate and evaluate multi-objective optimization models for the deployment of SN networks dedicated to the surveillance of sensitive sites. By considering a spectrum of objectives, including coverage maximization, connectivity, and cost-effectiveness, this research aims to develop methodologies that enhance the overall efficiency and effectiveness of SN network deployments in sensitive areas.

The present chapter offers a comprehensive analysis of WSNs in the context of multi-objective optimization. Clarifying the fundamentals of WSNs deployment is the main goal, especially as it relates to overcoming the difficulties involved in balancing several goals. In this chapter, we have focused on a detailed overview of WSNs.

2.2 Wireless Sensor Networks (WSNs)

WSNs operate as a distributed system comprising dedicated devices, including Sensor Nodes (SNs), Relay Nodes (RNs), and Sink Nodes. These devices collaborate to monitor various physical or environmental conditions, such as temperature, sound, pressure, and motion, across a defined area. As already discussed, SNs play a pivotal role by detecting environmental phenomena and converting them into electronic data. Following initial processing, this data is transmitted either directly to the Sink Node or routed through RNs. RNs are powerful devices in terms of energy and communication power, they are crucial in scenarios where direct communication between SNs and the Sink Node is impractical due to distance or physical obstructions. RNs facilitate the propagation of data from SNs by forming multi-hop paths, thereby extending the network's connectivity and ensuring the delivery of data to the intended Sink Node as shown in Fig 2.1 [7]. The Sink Node also referred to as a base station or gateway, collects data from all SNs, either directly or via RNs. Typically possessing greater computational power and storage capacity than SNs and RNs. The Sink Node may further forward the aggregated data to the users, a central system, or the cloud for in-depth analysis or decision-making.

In the operational sequence of WSNs, SNs collect data from the environment, and local processing may occur to reduce the transmitted data volume. Subsequently, this data is relayed, potentially traversing multiple RNs in a chain-like manner, until it reaches the Sink Node. The integration of RNs ensures data can be effectively transmitted over extended distances, and the network dynamically adapts to environmental changes or node failures, ensuring robustness and resilience [8].

WSNs can be categorized into homogeneous and heterogeneous networks based on the uniformity or diversity of nodes within the network. In a homogeneous network, all nodes share similar capabilities and functions, facilitating simpler management and coordination. On the other hand, heterogeneous networks encompass nodes with diverse capabilities, such as varying power levels, sensing capabilities, or communication range. While heterogeneous networks offer enhanced flexibility and efficiency, managing the diverse node characteristics presents additional challenges. The choice between homogeneous and heterogeneous architectures depends on specific application requirements, considering factors like energy efficiency, scalability, and the need for specialized functionalities [9].

WSNs are a collection of scattered autonomous SNs within a designated region that collects and transmits physical or environmental data to users, as illustrated in Fig 2.1. SNs, which are able to measure various physical qualities, often contain transceivers for data transmission and processing units. Wireless technologies like Bluetooth and Zigbee may allow these networks to operate without interruption in the absence of cable infrastructure. Communication between nodes is made possible by multi-hop networks ; data is sent across these nodes before returning to a base station or sink node.

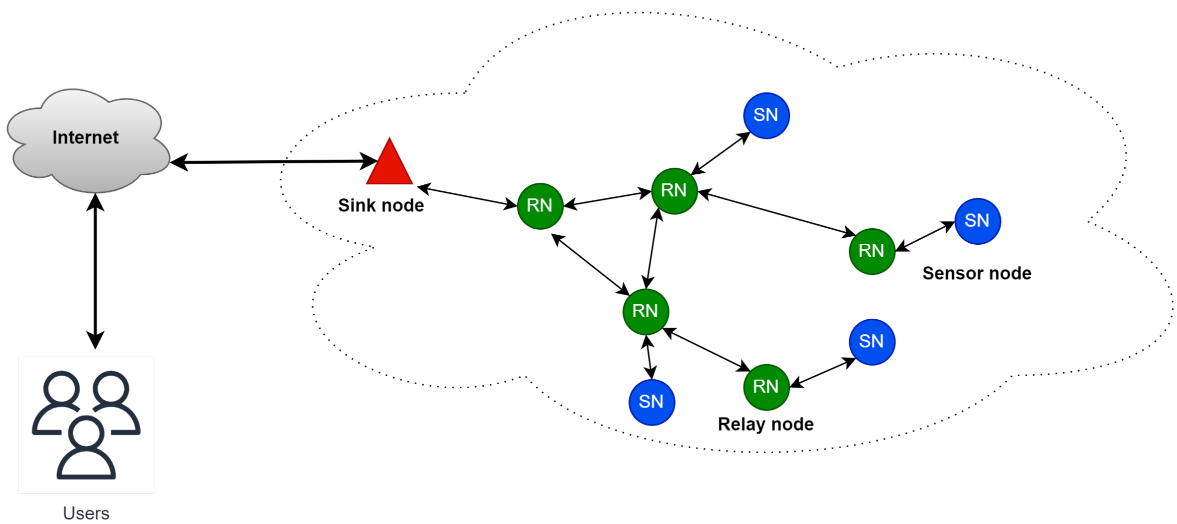


FIGURE 2.1 – A typical architecture of Wireless Sensor Networks (WSNs)

Specifically, energy-efficient protocols are required to extend the lifespan of networks, as SNs and RNs often run on batteries. This makes power management an important factor to take into account. Different topologies may be used by WSNs, including star, mesh, and cluster topologies. Cluster-based architectures with assigned cluster heads enable effective intra-cluster communication. Wide-ranging applications of WSNs may be found in a variety of fields, including as military operations, agriculture, healthcare, and environmental monitoring. WSNs confront a number of challenges, including scalability, security, and energy efficiency. Work is being done to design protocols that solve these challenges. In order to decide on data pathways and control access to the communication medium, routing and MAC protocols are essential. In general, WSNs offer a flexible and affordable way to gather data in a real-time, making a substantial contribution to the Internet of Things (IoT) environment [10].

Within WSNs, three main types of nodes play crucial roles : SNs, RNs, and Sink Nodes. Each node serves a distinct purpose in ensuring efficient data collection, transmission, and processing. Let us explore the definitions, roles, and architectures of these nodes to gain a comprehensive understanding of their contributions to the overall functionality of WSNs.

The role of SNs is to collect data from the environment using attached SNs, analyze and process the collected data locally, and transmit data to neighboring RNs, or directly to the sink node. Figure 2.2 shows the typical architecture of a SN.

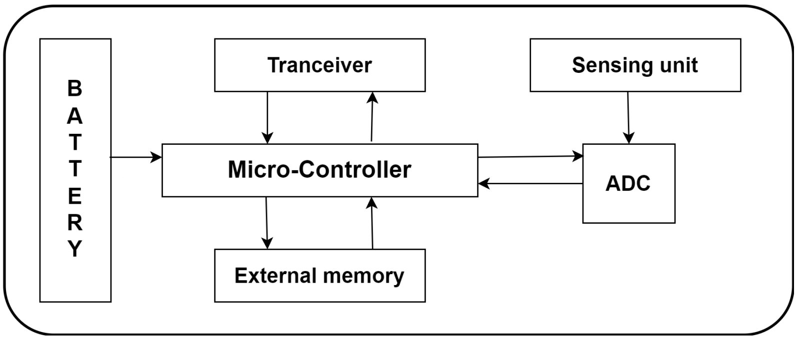


FIGURE 2.2 – A typical architecture of a Sensor Node (SN)

- **Sensing Unit :** is composed of various sensors for measuring physical parameters from the environment.
- **Communication Module (transceiver) :** facilitates wireless communication with other nodes.
- **Processing Unit (micro-controller) :** executes algorithms for data analysis and processing.
- **Memory :** stores readings from the sensors, processed data, and configuration information.
- **Power Supply :** is typically battery-powered or energy harvesting.

RNs, also known as a routers or intermediate nodes, are devices that help extend the communication range, overcome obstacles, and facilitate data transmission within WSNs. Thier main role is the relay of data between SNs and the sink node, extend the network's communication range, and help in creating alternative communication paths. Figure 2.3 shows the architecture of RN.

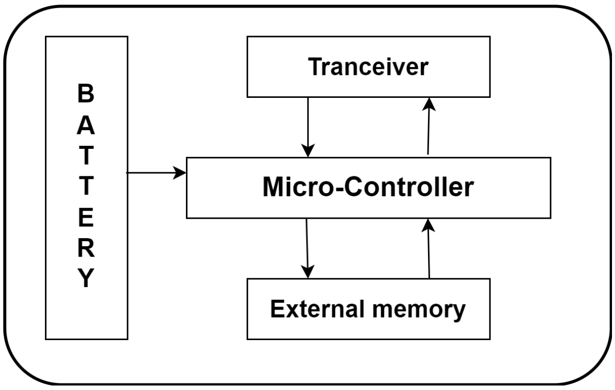


FIGURE 2.3 – A typical architecture of a Relay Node (RN)

- **Communication Module (transceiver) :** is used primarily on communication. It is equipped with a wireless transceiver (e.g., Zigbee, Wi-Fi, Bluetooth) to establish communication links with SNs, other RNs, or the central sink node.
- **Processing Unit (micro-controller) :** performs basic processing tasks, such as data aggregation or routing decisions.
- **Memory :** stores temporary data and routing information.
- **Power Supply :** is typically battery-powered or energy harvesting.

A sink node, also known as the base station or gateway, is a central node that collects, processes, and forwards data from the SNs to external networks or applications. Its role is to collect and aggregate data from multiple SNs and perform advanced processing and analysis of the aggregated data. A typical sink node consists of the following :

- **High-Performance Processing Unit :** is more powerful processing capabilities than SNs and RNs.
- **Extensive Memory :** stores aggregated data, historical information, and network parameters.
- **Communication Module :** connects a WSN to external networks or applications.
- **Power Supply :** is more stable power source compared to that of SNs and RNs.

2.3 Applications of WSNs

WSNs consist of spatially distributed autonomous SNs that monitor physical or environmental conditions. These SNs cooperatively pass their data through the network by means of RNs to a main location hosting the sink node. The potential applications of WSNs are vast, and they are being widely researched for various domains. Here are some notable applications :

Environmental Monitoring *:* WSNs play a vital role in environmental monitoring, focusing on ecosystems. Deployed in natural settings like forests and wildlife areas, WSNs track conditions such as temperature, humidity, air quality, and wildlife activities. In forests, WSNs strategically track endangered species, providing insights into migration, breeding, and potential threats. Real-time data aids conservationists, guiding strategies for ecosystem protection. WSNs combat illegal activities in protected areas by using motion detection. Prompt alerts enable swift responses, enhancing conservation measures.[11].

Infrastructure Health Monitoring : WSNs advance infrastructure health monitoring, focusing on critical structures like bridges and tunnels. Equipped with SNs, WSNs detect temperature changes, vibrations, and material strain. Continuous monitoring allows early issue identification, reducing unexpected failures and minimizing repair costs. WSNs create interconnected sensor networks, offering a holistic view for informed decision-making. Immediate alerts ensure timely maintenance. Integrating smart technologies augments resilience against environmental challenges. The synergy of WSNs and modern tech enhances structural safety and reliability.[12].

Border Security : WSNs enhance border security through interconnected SNs. WSNs detect motion, sound, and heat signatures, improving efficiency in monitoring challenging terrains. Comprehensive surveillance reduces blind spots, enhancing situational awareness. Real-time alerts in response to anomalies minimize the need for continuous human patrols. Integration with UAVs extends surveillance reach, creating a robust border security framework. The synergy of WSNs and modern surveillance technologies, including Relay Nodes (RNs), addresses evolving challenges in border control effectively. RNs play a crucial role in extending the communication range and ensuring seamless data transmission within the WSN, contributing to a more comprehensive and reliable border security system. [13].

2.3.1 Smart Car Park Surveillance :

The rapid growth of urban centers poses a significant challenge in efficiently managing and securing parking areas. To address this challenge, Smart Car Park Surveillance (SCPS), driven by WSNs, offers real-time monitoring and intelligent management of parking spaces.

In the context of a SCPS system, the parking area includes motion SNs. These nodes continuously monitor the occupancy status of individual parking slots by detecting the presence or absence of vehicles, allowing the system to accurately determine the real-time availability of parking spaces.

The use of WSNs in parking surveillance extends beyond simply detecting occupancy. SNs can identify unauthorized movements within the vicinity, enhancing the security framework. For example, the system can promptly notify management of a vehicle parked in a designated no-parking zone or detect unusual movements during off-hours, indicating potential security threats.

A crucial enhancement to this system involves integrating fire sensor nodes within the network. The heightened risk of fire in such environments requires early detection. Fire SNs promptly identify abnormal increases in temperature or the presence of smoke, triggering immediate alerts to management. In emergencies, these alerts activate critical safety protocols, such as activating sprinkler systems or notifying local fire departments.

Furthermore, the data collected by SCPS systems can be used to optimize parking space utilization. Analyzing occupancy patterns and demand allows urban planners to make informed decisions to enhance the overall efficiency of parking infrastructure. This benefits individual drivers by improving the availability of parking spaces and contributes to reduced traffic congestion, fostering a more sustainable urban environment.

In summary, the incorporation of WSNs in SCPS represents a technological advancement in strengthening both the security and efficiency of urban parking spaces. The real-time monitoring capabilities, coupled with intelligent data analysis, contribute to the development of a smarter and more responsive urban infrastructure, promoting safety, efficiency, and sustainability [14].

It should be noted that in this thesis the focus lies on optimizing WSNs deployment for smart car park surveillance.

2.4 Types of WSNs Deployment

The efficiency and effectiveness of WSNs heavily depend on the deployment strategy employed, as it directly influences network cost, coverage, connectivity, delay, and energy consumption. This section provides an in-depth exploration of two primary deployment types : *random* and *deterministic*.

2.4.1 Random Deployment

Random deployment is a widely used strategy in WSNs, characterized by the arbitrary placement of nodes within the area of interest. The primary advantage of random deployment lies in its simplicity and ease of implementation. This approach eliminates the need for intricate planning and enables rapid deployment, making it suitable for large-scale networks or areas with challenging accessibility.

One of the key challenges associated with random deployment is the non-uniform distribution of nodes. The randomness may lead to regions with high node density, causing potential redundancy and increased energy consumption, while other areas may suffer from sparse coverage, resulting in information loss or blind spots. To address this issue, researchers have proposed various optimization techniques, such as adjusting transmission ranges and utilizing mobility models to guide nodes towards optimal locations.

Moreover, random deployment is often accompanied by the phenomenon of node clustering, where nodes tend to aggregate in certain regions due to their random distribution. This clustering effect can be harnessed to enhance data aggregation and reduce communication overhead. However, it also introduces challenges related to load balancing and resource utilization.

2.4.2 Deterministic Deployment

In contrast to random deployment, deterministic deployment involves a systematic and planned placement of the WSN nodes based on specific criteria or optimization objectives. This deployment type aims to achieve a more uniform distribution of nodes across the area of interest, addressing the challenges associated with random deployment.

Deterministic deployment strategies can be broadly categorized into three main types : regular grid deployment, centroid-based deployment, and event-driven deployment.

Regular Grid Deployment : Regular grid deployment involves dividing the area of interest into a grid and placing SNs or RNs at regular intervals within the grid. This method ensures a uniform distribution of nodes, minimizes redundancy, and optimizes coverage. However, it may not be suitable for irregularly shaped areas, and adjustments may be required to accommodate obstacles or terrain variations.

Centroid-Based Deployment : Centroid-based deployment focuses on placing nodes around the centroid, or center of mass, of the area of interest. This approach aims to balance the distribution of nodes, providing more even coverage. Centroid-based deployment is particularly effective in irregularly shaped environments and can be adapted to address specific requirements, such as maximizing coverage along a particular axis.

Event-Driven Deployment : Event-driven deployment is tailored to the specific characteristics of the monitored environment. WSNs nodes are strategically placed based on the expected occurrence of events or phenomena. For example, in environmental monitoring, nodes may be deployed around known pollution sources or in areas prone to natural disasters. This type of deployment optimizes resource utilization by focusing nodes on areas of interest.

Deterministic deployment strategies often require more planning and computational resources compared to random deployment. However, the benefits include improved cost, coverage, connectivity, delay, reduced energy consumption, and enhanced network reliability. Researchers have developed various optimization algorithms to aid in the planning and execution of deterministic deployment strategies.

The choice between random and deterministic deployment depends on the specific requirements and characteristics of the WSN application. Random deployment offers simplicity and rapid deployment, making it suitable for scenarios where quick setup is essential. However, the challenges of non-uniform node distribution and potential clustering effects must be carefully considered.

Deterministic deployment, on the other hand, provides a more controlled and optimized approach, minimizing redundancy and ensuring better coverage. The trade-off, however, involves increased planning complexity and computational overhead during deployment. Researchers continue to explore hybrid deployment strategies that combine the advantages of both random and deterministic approaches to achieve a balance between simplicity and optimization.

In conclusion, the deployment strategy plays a pivotal role in determining the performance and efficiency of WSNs. Random deployment offers simplicity and speed, while deterministic deployment aims for optimization and uniformity. The choice between these strategies depends on the specific application requirements, environmental characteristics, and the trade-offs that align with the goals of the WSN deployment. Ongoing research and advancements in deployment techniques contribute to the evolution of WSNs, ensuring their adaptability to diverse scenarios and enhancing their effectiveness in various real-world applications.

2.5 Sensing Models in WSNs

Coverage is a pivotal criterion in the efficacy of WSNs, ensuring that every significant point or event within an area of interest is under the surveillance of at least one SN. Among various models that conceptualize this feature, the Binary and Probabilistic coverage models are notably prominent. Both models offer distinct perspectives on how SNs detect events in their environment.

2.5.1 Disk Sensing Model

The Disk Sensing Model operates under a dichotomous paradigm. As depicted the figure 2.4, an SN, within its predefined sensing range R_s, can unequivocally detect the presence of an event or phenomenon. Beyond this range, its detection capability drops to zero.

Characteristics :

- Fixed Sensing Range : Each SN possesses a fixed and definitive sensing range. Anything within this range is detected with certainty, while events beyond remain undetected.

— Clear Boundaries : There is a distinct boundary that demarcates the covered and uncovered areas.

— Uniform Detection : Within the sensing range, the detection capability is uniform. That is, the likelihood of detection is the same throughout the entire range.

Modeling : When an event occurs, at a distance r from an SN, the detection probability is as follows :

$$P_{detect}(r) = \begin{cases} 1, & \text{if } r \leq R_s \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

As illustrated in Figure 2.4, the event 1, located within a distance $r1 \leq R_s$, is effectively detected, while the event 2 positioned at a distance $r2 \geq R_s$ remains undetected.

Applications : Given its simplistic nature, the Binary Model is ideal for applications where the demarcation between covered and uncovered regions needs to be explicit, such as perimeter monitoring or flood detection.

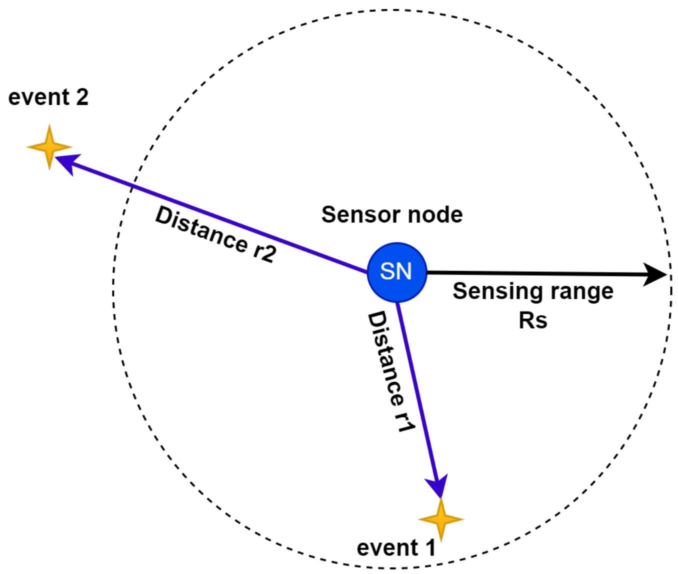


FIGURE 2.4 – The disk sensing model

2.5.2 Probabilistic Sensing Model

In stark contrast to the binary model, the Probabilistic Coverage Model incorporates the probability associated with detecting an event. This probability can vary based on several factors, including the distance from the SN, external interferences, and SN quality [15].

Characteristics :

— Variable Detection Probability : The likelihood of an event being detected is not constant across the sensing range. Generally, as an object moves farther from the SN, the probability of detection diminishes.

— Incorporates SN Imperfections : Real-world SNs are not flawless. This model acknowledges possible SN malfunctions, inaccuracies, or degradation over time.

— Environmental Influences : Factors such as atmospheric conditions, obstructions, or interference can affect the detection probability.

Modeling : The detection probability often follows a Gaussian distribution or other suitable distributions. For instance, if $P_{detect}(r)$ denotes the detection probability at distance r from the SN, then $P_{detect}(r)$ might be higher when r is small and decreases as r increases. The detection probability is as follows :

$$P_{detect}(r) = \begin{cases} e^{(-\beta.r)}, & \text{if } r \leq R_s \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

The parameter β is linked to the sensor's physical attributes and can be determined through field experiments conducted for the particular sensor network in question.

Applications : The Probabilistic Model shines in more complex environments where variability and uncertainty are inherent. This includes wildlife tracking in variable terrains, pollution level monitoring in urban areas, or any scenario where environmental factors can influence SN readings.

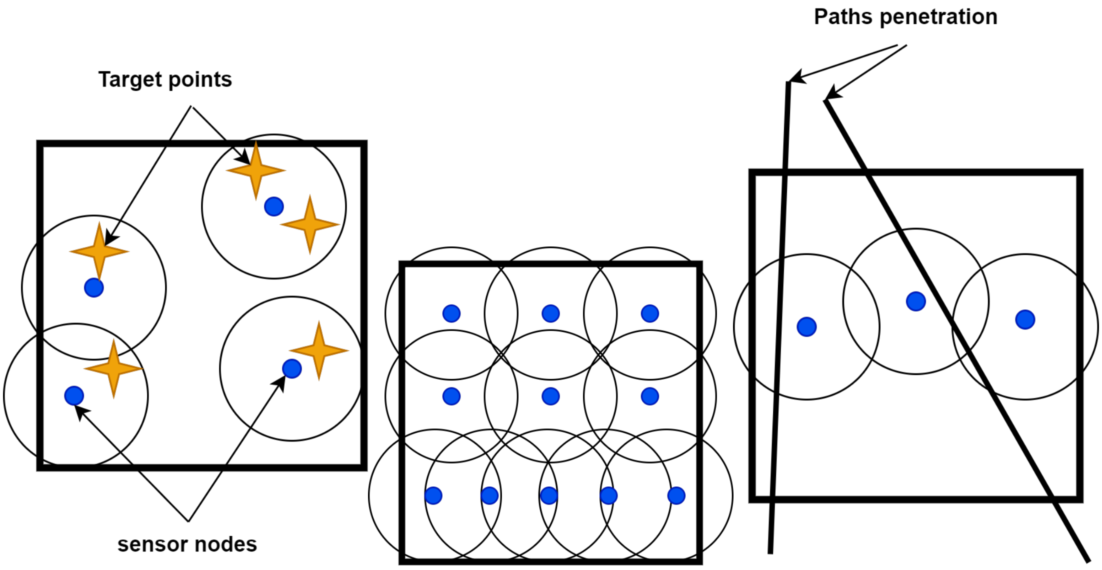
2.6 Coverage in WSNs

In the realm of WSNs, coverage is a critical parameter that reflects the quality of service of the network in terms of monitoring and sensing [16]. Depending on the application's objectives, different types of coverage become more pertinent. The three main types of coverage in WSNs are target Point Coverage, Area Coverage, and barrier Coverage. As an illustrative example, Figure 2.5 showcases the three varieties of coverage.

Target Point Coverage : As shown Figure 2.5a, in this type of coverage, the primary focus is on individual points of interest within the sensing field. The objective is to ensure that each of these points is within the sensing range of at least one SN. This coverage type is critical in scenarios where there are specific locations or assets that need continuous monitoring, such as monitoring machinery in a factory or surveillance of fixed assets. The challenge is that Ensuring continuous coverage to specific points might require strategic placement of SNs or frequent recalibration, especially if there is a risk of SN failures [17].

Area Coverage : As the name suggests and Figure 2.5b illustrates, the focus here is to cover an entire region or area, ensuring that every event occurring within this region is detected by at least one SN. This is most relevant for environmental monitoring applications like forest fire detection, where any event within a specified region needs to be sensed. Another use case is agricultural fields where the health and conditions of crops over a vast area need to be monitored. The challenge is that achieving full area coverage can be resource-intensive as it might require a dense deployment of SNs. Balancing between energy consumption and coverage redundancy becomes crucial [15].

Barrier Coverage : In Figure 2.5c, the presentation demonstrates barrier coverage, which is designed to identify and impede the movement of an entity traversing a belt or barrier. Here, the objective is not to monitor every point in the field but to ensure no object can pass through the barrier without being detected. It is predominantly used in border surveillance and intrusion detection where the main goal is to detect unauthorized crossings . The challenge is while it might require fewer SNs than full area coverage, the challenge lies in ensuring there are no gaps in the barrier that might allow undetected crossings [15].



(A) (B) (C)

FIGURE 2.5 – Types of coverage : (A) Target point coverage (B) Area coverage (C) barriere coverage

In conclusion, the type of coverage necessary for the WSN largely depends on the intended application. A proper understanding of these classifications aids in optimizing SN deployment, ensuring efficient resource utilization, and achieving desired monitoring outcomes.

2.7 Communication Channel Models in WSNs

In WSNs, the optimization of deployment strategies is a critical research facet, necessitating a profound comprehension of communication channel characteristics. Communication Channel Models serve as indispensable analytical tools, encapsulating the intricate facets of signal propagation, interference mitigation, and energy consumption. A robust understanding of these models is imperative for strategically placing sensor nodes, thereby ensuring reliable communication and efficient data transfer within the network. This study delves into key communication channel models that underpin deployment optimization, commencing with the foundational Boolean Disk Model. While seemingly elementary, the Boolean Disk Model serves as an essential starting point, offering a simplified representation of communication range. Its simplicity provides a clear baseline, motivating the exploration of more sophisticated models.

The subsequent delineation of the probability model, propagation model, interference model, and radio energy model expands upon the boolean disk model, each contributing nuanced insights and analytical depth to inform deployment optimization strategies within WSNs. This academic pursuit seeks to enrich the discourse surrounding the WSN deployment methodologies, offering a comprehensive understanding of communication channel models. The choice of a communication model in WSNs depends on factors such as energy efficiency requirements, scalability, and the nature of the monitored environment, allowing for adaptability to diverse applications and scenarios [18]. To describe various communication scenarios and parameters affects the transmission, various models are used as explained below :

2.7.1 Disk Communication Model

A basic yet powerful model, the Boolean disk model dictates that a SN can communicate with all others within a fixed range, r, represented as :

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \leq r \tag{2.3}$$

where :

— (x_1, y_1) and (x_2, y_2) : are the coordinates of two nodes.

— r : is the communication range.

2.7.2 Probability Model

In this model, the probability of successful communication between two nodes diminishes as their separating distance increases. A frequently used probability function in this context is :

$$P(d) = \frac{1}{1 + (\frac{d}{d_0})^\gamma} \tag{2.4}$$

where :

— $P(d)$: is the probability of successful communication over distance d.

— d_0 : is the reference distance.

— γ : is the path loss exponent.

2.7.3 Propagation Model

The path loss or propagation model in wireless communication describes the attenuation of signal strength as it travels from the transmitter to the receiver. This model is crucial for predicting signal strength, coverage, and link quality in wireless communication systems, including WSNs. Several propagation models exist, each suitable for different environments and scenarios. Two common types are free-space path loss and log-distance path loss models [19].The log-distance path loss model, a popular choice in WSNs communications, mathematically represents the path loss as :

$$PL(d) = PL(d_0) + 10 \cdot n \cdot \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \tag{2.5}$$

where :

— $PL(d)$: is the path loss at distance d.

— $PL(d_0)$: is the path loss at a reference distance d_0.

— n : is the path loss exponent, indicating the rate at which the path loss increases with distance.

— X_σ : is a Gaussian random variable representing the shadowing effects.

2.7.4 Interference Model

The Signal to Interference plus Noise Ratio (SINR) serves as a critical metric in the realm of wireless communications, offering insights into the impact of signal interference and background noise. This model, expressed by the formula *SINR* as given below, encapsulates the delicate balance required for effective wireless transmission. Here, P_s represents the power of the intended signal, embodying factors like transmission strength and propagation characteristics. On the other hand, P_i accounts for interference originating from simultaneous transmissions, introducing the potential for signal distortion. Meanwhile, P_n quantifies the background noise power, encompassing extraneous factors that contribute to signal degradation [20].

The SINR acts as a pivotal determinant of communication quality, with a higher ratio indicating a more favorable signal-to-noise and interference environment. Engineers and network designers leverage this model to optimize system parameters, including transmitter power and channel allocation, aiming to mitigate interference effects and enhance overall communication reliability. In practical wireless scenarios, achieving an optimal SINR is paramount for ensuring low error rates and robust communication performance. The Signal to Interference plus Noise Ratio (SINR) is given by :

$$SINR = \frac{P_s}{P_i + P_n} \tag{2.6}$$

where :

— P_s : is the signal power.

— P_i : is the interference power from other simultaneous transmissions.

— P_n : is the background noise power.

2.7.5 Radio Energy Model

The radio energy model provides a quantitative framework for assessing the energy consumption associated with data transmission and reception in wireless communication systems. This model is particularly crucial for WSNs, where energy efficiency is a critical consideration due to the limited power resources of SNs [21]. The energy consumption for both transmission and reception is expressed as a function of the message length of k-bit and the transmission distance d. This model reperesents the energy consumption during data transmission and reception. The energy to transmit and receive a k-bit message over a distance d can be represented as :

$$E_{TX}(k,d) = E_{elec} \cdot k + \varepsilon_{amp} \cdot k \cdot d^2 \tag{2.7}$$

$$E_{RX}(k) = E_{elec} \cdot k \tag{2.8}$$

where :

— E_{elec} : is the energy consumed per bit to run the transmitter or receiver.

— ε_{amp} : is the energy consumed by the amplifier.

.

2.8 Routing in WSNs

WSNs rely on effective routing methods to facilitate the efficient exchange of data among resource-constrained SNs. Various routing strategies have been devised to address the unique challenges inherent in WSNs, where energy conservation and network longevity are paramount. Flat routing, treating all nodes equally, and hierarchical routing, organizing nodes into clusters with designated cluster heads, are fundamental approaches. Multipath routing introduces redundancy by establishing multiple paths between nodes, ensuring both load balancing and fault tolerance. Location-based routing utilizes geographical information to make informed routing decisions, enhancing efficiency [22].

Data-centric routing, focusing on content rather than node addresses, and QoS-aware routing, prioritizing certain service levels, cater to specific application needs. Cross-layer routing fosters collaboration between different protocol layers for more nuanced decision-making. Energy-efficient routing protocols play a critical role in maximizing the lifespan of WSNs through strategies like dynamic sleep scheduling and energy-efficient data aggregation. Ultimately, the choice of routing method depends on the specific requirements of the WSN application, reflecting the ongoing efforts to strike a balance between energy efficiency, scalability, and reliable data delivery in dynamic and resource-constrained wireless environments. Mainly, the routing can be classified into single-tier and two-tier architecture as explained below.

2.8.1 Single-Tiered Routing

In a single-tiered routing, both SNs and RNs participate in the forwarding of data packets. Here, SNs are primarily responsible for detecting events or measuring certain values in their environment. Once they gather this data, they not only send their sensed data but can also relay data packets from other SNs toward a destination, typically a sink or base station. This creates a flexible and interconnected network where each node can serve as a potential forwarding point for other nodes' data. The advantage is the provision of multiple paths for data transmission, which can increase the resilience and robustness of the network. However, a potential drawback is the increased energy consumption for SNs since they are engaged in both sensing and data forwarding, which could reduce the overall network lifespan[23]. For example, as shown in Figure 2.6, upon detecting an event within the operational vicinity of an SN, it initiates the creation of a data packet to forward it to the designated sink node. The inherent limitation of the SN's communication range necessitates strategic routing decisions. In this context, the sensor node, aware of its restricted communication range, identifies the RN within its immediate neighborhood. The generated packet is consequently transmitted to this proximate relay node, serving as the next hop. Subsequently, the relay node dutifully relays the packet to a next hop SN, initiating a cascading sequence involving two subsequent relay nodes situated along the path leading to the ultimate destination—the sink node. It should be noted that the given routing path contains both SNs and RNs, which is the definition of the single-tiered architecture.

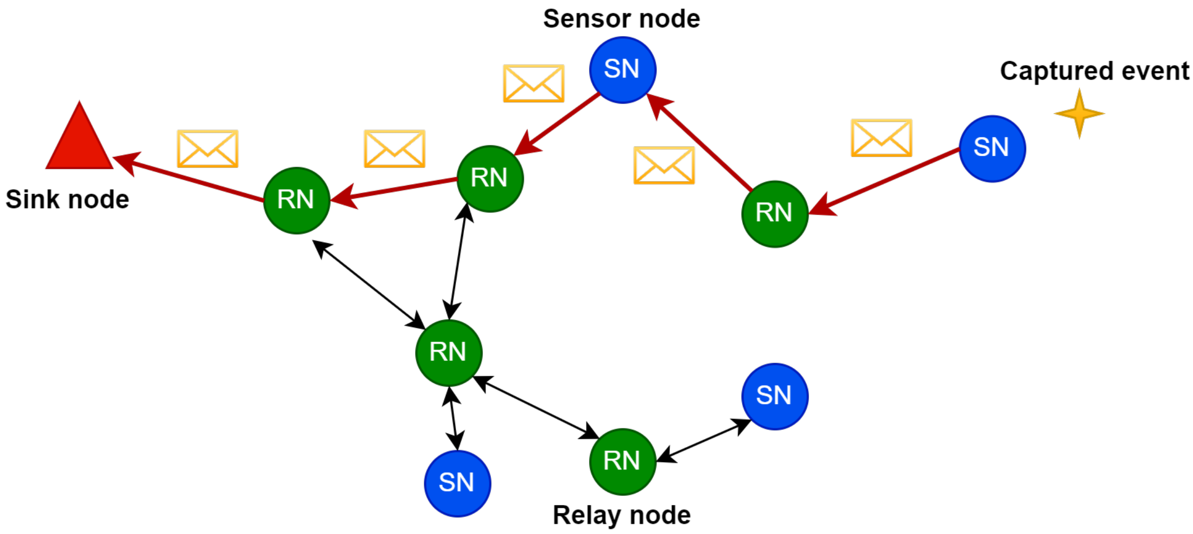


FIGURE 2.6 – An example of a single-tiered WSN.

2.8.2 Two-Tiered Routing

In the two-tiered routing, there is a more defined distinction in the roles of the nodes. SNs are responsible for sensing data from the environment. Once they have this data, they transmit it to a RN or directly to a sink node. These RNs, as the name suggests, have the primary role of relaying or forwarding this data to the final destination. They do not engage in sensing but act as intermediaries, taking data from SNs and ensuring its safe passage to the sink node. This division of roles allows SNs to conserve energy since they only need to transmit data over short distances and do not handle data forwarding from other nodes. However, the network might face challenges if RNs become overloaded or if there are not enough of them, leading to potential communication bottlenecks [24]. As depicted in Figure 2.7, the distinction between the two-tiered architecture and the single-tiered configuration becomes apparent, emphasizing the exclusive involvement of RNS in the routing path. In this illustrative scenario, the operational dynamics unfold when an SN perceives an event within its range. In response, the SN initiates the creation of a data packet containing information about the detected event. Subsequently, the SN strategically identifies a nearby RN within its vicinity. This identified RN becomes the immediate next hop for the data packet.

Effectively functioning as the next hop in the communication sequence, the selected RN receives the packet from the SN. Now, the RN assumes the responsibility of transmitting the packet to the next RN in the communication path. This routing process continues iteratively, with the packet successively passing through two more RNs along the designated path. The sequence culminates when the final RN directs the packet to its ultimate destination—the sink node.

In essence, this two-tiered architecture distinguishes itself by relying exclusively on RNs to facilitate the transfer of information. The visual representation in Figure 2.7 encapsulates the step-by-step progression from the SN initiating the packet creation to its final delivery at the sink node through a series of RNs.

It is worth mentioning that the present research sutdy assumes a two-tiered architecture to minimize network delay and energy consumption.

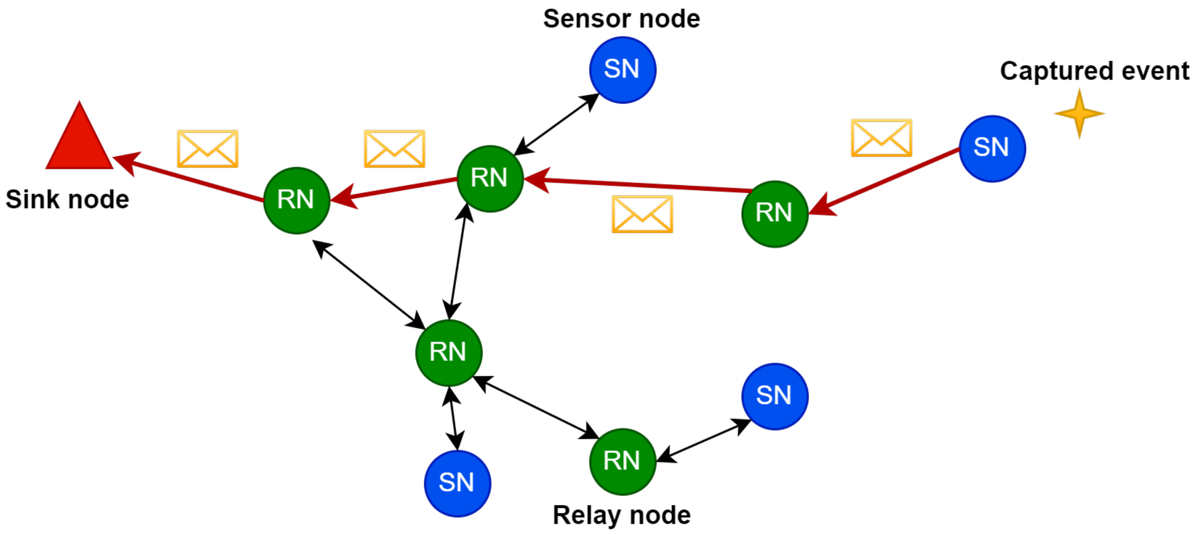


FIGURE 2.7 – An example of a two-tiered WSN.

2.9 Objectives and Requirements of WSNs Deployment

WSNs are deployed for various objectives, catering to a range of applications in diverse fields such as environmental monitoring, healthcare, industrial automation, and smart cities. One primary objective is to enhance data collection and monitoring capabilities in environments where wired infrastructure is impractical or cost-prohibitive. WSNs facilitate real-time data acquisition from distributed SNs, enabling efficient monitoring of physical phenomena. For instance, in environmental monitoring, WSNs can be deployed to collect data on temperature, humidity, and pollution levels in remote or inaccessible locations. Additionally, WSNs contribute to improving efficiency and automation by enabling seamless communication and control in industrial settings, optimizing processes and reducing manual intervention [25].

The deployment of WSNs is guided by specific requirements that vary based on the intended application. Common requirements include energy efficiency, as many SNs are battery-powered and may be deployed in challenging environments. Network scalability is crucial to accommodate a varying number of SNs, and reliability is paramount to ensure the delivery of accurate and timely data. Security is another critical requirement, especially in applications like healthcare or military, where the integrity and confidentiality of data are of utmost importance. Furthermore, WSNs may need to operate in dynamic and harsh environments, requiring robust and resilient communication protocols. Balancing these objectives and requirements is essential to design WSNs that effectively meet the demands of their intended applications [26].

Network Cost : Network cost in the WSN involves the financial outlay associated with deploying and maintaining the network infrastructure. This includes the expenses related to procuring SNs, communication modules, base stations, energy sources, and any necessary supporting infrastructure. Efficient management of network costs is crucial for the economic viability and sustainability of WSNs, particularly in applications where cost-effectiveness is a primary concern [27].

Network Delay : Network delay in WSNs refers to the time taken for data to travel from source SNs to destination nodes or sink node. It encompasses various components, including propagation delay, transmission delay, queuing delay, and processing delay. Minimizing network delay is crucial in applications that require real-time data, such as healthcare or industrial automation, to ensure timely decision-making based on the collected information. Minimizing the hop-count, or the etwork diameter, in WSNs reduces the number of intermediate nodes that data packets need to traverse, thereby lowering transmission delays. This optimization shortens the path length between nodes, enhancing communication efficiency and reducing packet delivery time, ultimately aligning with the goal of minimizing network delay.

Target Point Coverage : Target point coverage in WSNs pertains to the extent of the target points covered by the deployed SNs. Achieving optimal target point coverage involves strategically placing nodes to ensure comprehensive data collection and monitoring. This aspect is particularly important in applications like environmental monitoring or surveillance, where coverage of the target points is essential for accurate and meaningful data analysis [15].

Network Connectivity : Network connectivity in WSNs refers to the ability of SNs to establish and maintain communication links within the network. Reliable connectivity is vital for seamless data transmission and collection. Achieving and maintaining network connectivity involves addressing challenges such as node mobility, environmental obstacles, and potential signal interference. Robust network connectivity is critical for the overall effectiveness of WSNs in various applications, including those in dynamic or challenging environments.

2.10 Summary

This chapter has described various aspects of WSNs, detailing their essential components, including SNs and SNs and their internal architectures. The chapter has explored diverse applications of the WSN, ranging from environmental monitoring to smart car park surveillance. The aim was to set the stage for a comprehensive understanding of the technology's real-world implications. The chapter also discussed communication models, routing, and deployment types within the WSN. Various coverage models were discussed, including binary and probabilistic coverage, as well as different coverage types such as target point, area, and barrier coverage. The chapter then described the different deployment objectives and requirements of WSNs, emphasizing factors such as network cost, delay, target coverage, and connectivity.

Chapitre 3

Overview of Combinatorial Optimization

3.1 Introduction

Combinatorial optimization plays a crucial role in addressing the complexities associated with the allocation of resources and strategic placement of SNs and RNs in the context of the multi-objective optimization for the deployment of WSNs. This problem involves simultaneously optimizing multiple conflicting objectives, such as maximizing coverage, minimizing deployment costs, and ensuring robustness to potential threats. Combinatorial optimization techniques provide a systematic and efficient approach to exploring the vast solution space, considering various combinations of SNs and RNs placements and configurations [28].

One key aspect is the strategic selection of node locations to maximize the coverage of the sensitive site while minimizing the number of deployed nodes and associated costs. This inherently involves a combinatorial decision-making process where each potential node location represents a discrete choice. Additionally, the multi-objective nature of the optimization requires balancing trade-offs between conflicting goals, such as achieving extensive surveillance coverage and maintaining a cost-effective deployment. Advanced combinatorial optimization algorithms, including genetic algorithms, simulated annealing, or multi-objective evolutionary algorithms, can be employed to efficiently explore the solution space, providing a set of trade-off solutions known as the Pareto front [29]. Ultimately, by leveraging combinatorial optimization techniques [28], the deployment of WSNs for sensitive site surveillance can be systematically optimized to meet diverse and often conflicting objectives. These methodologies contribute to enhancing the overall effectiveness, efficiency, and resilience of WSNs deployments in safeguarding critical infrastructure and sensitive locations. Additionally, the integration of these combinatorial optimization algorithms addresses the inherent algorithmic complexity, ensuring the scalability and computational efficiency crucial for real-time surveillance applications in dynamic environments.

3.2 Algorithmic Complexity

The algorithmic complexity of multi-objective optimization for the deployment of WSNs is characterized by the inherent computational challenges associated with simultaneously optimizing conflicting and diverse objectives. This complexity stems from the need to explore a multi-dimensional solution space, where each potential deployment configuration represents a trade-off between different surveillance objectives. The presence of multiple conflicting criteria, such as maximizing coverage, minimizing deployment costs, and ensuring robustness, adds intricacy to the optimization process [30].

The majority of algorithms used for multi-objective optimization in WSNs deployment are heuristic or metaheuristic in nature. These algorithms, such as evolutionary algorithms (e.g., NSGA-II, SPEA2), simulated annealing, and particle swarm optimization [30], are designed to explore solution spaces and identify Pareto-optimal solutions efficiently. The time complexity of these algorithms is often expressed in terms of the number of iterations or generations, and they provide approximate solutions that are close to the true Pareto front in a reasonable amount of time. While these heuristic approaches do not guarantee optimality, they offer scalability advantages, making them suitable for large and complex WSNs deployment scenarios.

3.2.1 Complexity Classes

Complexity classes are a fundamental concept in theoretical computer science that categorizes problems based on the computational resources required to solve them. These classes provide a framework for understanding the inherent difficulty of computational problems and the efficiency of algorithms in solving them. The most prominent complexity classes include P (Polynomial Time), which consists of problems solvable in polynomial time, and NP (Non-deterministic Polynomial Time), representing problems for which a potential solution can be verified in polynomial time[31].

The complexity class P consists of problems that can be solved in polynomial time. The mathematical equation represents P as the union of time-bounded classes, where each class corresponds to a polynomial of increasing degree.

$$P = \bigcup_{k=1}^{\infty} TIME(n^k) \tag{3.1}$$

NP represents problems for which a potential solution can be verified in polynomial time. The mathematical expression denotes NP as the union of non-deterministic time-bounded classes.

$$NP = \bigcup_{k=1}^{\infty} NTIME(n^k) \tag{3.2}$$

If a language L is both in NP and any problem in NP can be reduced to L in polynomial time, then L is NP-complete. The LaTeX code represents this by stating that language L is in the class NP-Complete.

$$L \in NP - Complete \tag{3.3}$$

$$\begin{aligned} 1. \quad & L \in \text{ NP} \\ 2. \quad & \forall A \in \text{ NP, there exists a polynomial-time computable function } f \\ & \text{such that for every instance } x \text{ of } A, \\ & x \text{ is a yes-instance of } A \iff f(x) \text{ is a yes-instance of } L \end{aligned}$$

NP-complete is a subset of NP encompassing problems that are as hard as the hardest problems in NP, making them crucial in the study of computational complexity.

Additionally, there are complexity classes like PSPACE, which considers problems that can be solved using polynomial space, and EXP, representing problems solvable in exponential time.

These classes enable researchers to classify problems based on their computational characteristics, providing insights into the boundaries of efficient computation and the limitations of algorithmic solutions.

Understanding complexity classes is vital for addressing questions related to the inherent difficulty of problems and the capabilities of algorithms. The concept of completeness within these classes, such as NP-complete problems, plays a crucial role in theoretical computer science as it provides a lens through which researchers can analyze the relationships between different problems and assess their computational complexity. The study of complexity classes not only aids in developing efficient algorithms but also contributes to solving practical computational challenges and assessing the feasibility of solving complex problems within realistic resource constraints.

3.3 Complexity in the WSN Deployment

In the deployment of WSNs, complexity arises from the intricate set of challenges associated with strategically situating SNs to achieve specific surveillance or monitoring objectives. Spatial complexity is a fundamental aspect, encompassing the task of identifying optimal locations for SN placement. This involves addressing geographical constraints, considering the spatial distribution of targets, and optimizing coverage to minimize blind spots. The spatial complexity also extends to the dynamic nature of the deployment area, requiring adaptive strategies to account for changes in the environment or potential obstacles [32].

Resource complexity involves managing the limited resources of individual SNs within the network. Efficient deployment necessitates balancing considerations of power consumption, bandwidth utilization, and memory constraints to ensure the longevity and sustained functionality of the WSN. Designing deployment strategies that optimize resource usage while meeting the operational requirements of the surveillance task is a crucial aspect of mitigating resource-related complexities. Additionally, complexities arise in handling and processing the data generated by the deployed WSNs, requiring effective algorithms for data fusion and analysis to extract meaningful insights from the collected information. These multiple dimensions of complexity collectively influence the effectiveness and sustainability of WSNs deployments in diverse operational environments.

3.4 Combinatorial Optimization Problems

Combinatorial optimization problems play a crucial role in the efficient design and operation of WSNs, which consist of spatially distributed SNs communicating wirelessly to monitor and collect data from the surrounding environment. One significant problem in WSNs is the SN deployment problem. Given a geographical area to be monitored, the challenge is to determine the optimal locations for SN placement to achieve maximum coverage while minimizing the number of SNs deployed. This problem involves a combinatorial aspect, as the decision involves selecting a subset of possible locations for SN installation, each with associated trade-offs between coverage and deployment costs [28].

Another important problem in WSNs is the energy-efficient routing problem. SNs in WSNs are typically powered by limited energy sources, and optimizing energy consumption is crucial for prolonging the network's lifetime. This combinatorial optimization problem involves finding the most energy-efficient routes for data transmission from the SNs to a designated sink or base station. Balancing the trade-off between energy efficiency and communication reliability is a critical aspect of solving this problem, where the selection of optimal routes and transmission schedules becomes a combinatorial decision.

Moreover, the data aggregation Problem in WSNs is a combinatorial optimization challenge. Aggregating this information before transmission can significantly reduce communication overhead and energy consumption in scenarios where SNs collect redundant or correlated data. This problem involves determining an optimal schedule for aggregating and transmitting data while considering available energy and network connectivity constraints. Combinatorial optimization techniques are essential for efficiently exploring the solution space and identifying the best combinations of SNs and data aggregation points to enhance the overall performance and longevity of WSNs [33].

3.5 Multiobjective Optimization Problems

Formulating a multiobjective optimization problem involves balancing conflicting goals to achieve an optimal trade-off between cost, hop-count, coverage, and connectivity. The objective is to minimize deployment costs and hop-count while simultaneously ensuring comprehensive coverage and robust network connectivity. The optimization problem can be expressed mathematically with objective functions representing the minimization of both cost and hop-count. Constraints are introduced to maintain coverage by strategically placing SNs and adjusting their sensing ranges, while also guaranteeing connectivity through the optimization of routing algorithms and the inclusion of redundancy mechanisms[34]. This holistic approach addresses the challenges of achieving cost-effectiveness and efficient data transmission within WSNs while maintaining the essential requirements of coverage and connectivity.

In this optimization scenario, the challenge is to find a deployment strategy that not only minimizes costs and hop count but also guarantees coverage and connectivity, which are crucial for the effectiveness of the WSN. The interplay of these objectives requires careful consideration of trade-offs, as reducing costs or hop-count may potentially compromise coverage or connectivity. Thus, the optimization process involves finding a Pareto-optimal solution, considering the inherent trade-offs between conflicting objectives, to strike an optimal balance and ensure a well-performing and cost-efficient WSNs deployment.

3.5.1 Mathematical Formulations for MOPs

Multi-objective optimization problems (MOPs) fall within the realm of combinatorial optimization. While some definitions are influenced by combinatorial optimization, distinct concepts unique to multi-objective optimization are introduced. The primary specificity of multi-objective optimization lies in the existence of multiple functions to optimize. Consequently, it is essential to reconsider the concept of solution optimality to accommodate the presence of several objectives.

$$\text{(MOP)} \quad \text{Optimize } F(x) = (f_1(x), f_2(x), \ldots, f_n(x)) \tag{3.4}$$

subject to $x \in D \quad Eq(3.4)$, where n is the number of objectives ($n \geq 2$), $x = (x_1, x_2, \ldots, x_k)$ is the vector representing decision variables, D represents the set of feasible solutions, and each of the functions $f_i(x)$ is to be optimized, i.e., minimized or maximized. In the present discussion, the main focus will be exclusively on minimization problems within the context of multi-objective optimization. The primary challenge inherent in multi-objective problems lies in the absence of a singular definition for an optimal solution. The decision-maker can merely express a preference for one solution over another, but there is not a single solution superior to all others. Consequently, the resolution of a multi-objective problem does not center around identifying an optimal solution; rather, it involves determining a set of satisfactory solutions for which no straightforward ranking can be established. The methodologies employed for solving multi-objective problems, therefore, function as decision support tools, as the ultimate decision is delegated to the decision-maker [35].

To tackle this challenge, the scientific community has adopted two distinct approaches. The first involves simplifying a multi-objective problem into a single-objective problem, albeit at the potential cost of losing the intricacies of the original problem. The second approach aims to address the problem by taking into account all relevant criteria. This segment of the scientific community has introduced numerous innovative resolution methods in recent years.

Definition Pareto Dominance : It is said that the vector $x \in D$ dominates the vector $y \in D$ if :

$$f_i(x) \leq f_i(y), \forall i \in \{1,2,\ldots,n\}, \exists j \in \{1,2,\ldots,n\} \text{ such that } f_j(x) < f_j(y) \tag{3.5}$$

It is denoted as $x \succcurlyeq y$ and read as : x Pareto dominates y.

A general form for a Pareto Multi-Objective Optimization problem (P-MOP) can be represented as follows :

General Formulation :

$$\begin{aligned} \text{Minimize or Maximize} \quad & \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})] && (3.6)\\ \text{Subject to :} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1,2,\ldots,m && (3.7)\\ & h_j(\mathbf{x}) = 0, \quad j = 1,2,\ldots,p && (3.8)\\ & \mathbf{x} \in \mathcal{X} && (3.9) \end{aligned}$$

where :

— $\mathbf{x}$ represents the decision vector.

— $\mathbf{f}(\mathbf{x})$ is a vector of k objective functions to be minimized or maximized.

— $g_i(\mathbf{x})$ and $h_j(\mathbf{x})$ are inequality and equality constraints, respectively.

— m is the number of inequality constraints, and p is the number of equality constraints.

— $\mathcal{X}$ is the feasible solution space.

The goal is to find a set of solutions $\mathbf{x}^*$ in the decision space $\mathcal{X}$ such that no other feasible solution $\mathbf{x}'$ exists for which $\mathbf{f}(\mathbf{x}')$ is superior or equal to $\mathbf{f}(\mathbf{x}^*)$ in all objectives, and at least one objective value is strictly better.

Example P-MOP Problem : Consider a specific example where you want to minimize both the cost (C) and the hop-count (H) in a WSN while maintaining coverage (CO) and connectivity (CN). This could be formulated as :

Minimize :

$$\mathbf{f}(\mathbf{x}) = [C(\mathbf{x}), H(\mathbf{x})] \tag{3.10}$$

Subject to :

$$CO(\mathbf{x}) \geq \text{Threshold Coverage}$$
$$CN(\mathbf{x}) \geq \text{Threshold Connectivity}$$

Solving such PMO problems involves finding solutions on the Pareto frontier, representing the trade-off between minimizing cost and hop-count while satisfying coverage and connectivity constraints. Various algorithms, such as evolutionary algorithms or swarm intelligence methods, can be employed for Pareto-based multi-objective optimization [36].

3.5.2 Resolution Methods for MOPs

Solving a multi-objective problem presents a distinctive challenge compared to a single-objective scenario, as it entails not just a single, unique solution but an entire set known as the Pareto Front. This concept introduces the need for a decision-maker. In light of this, resolution methods for such problems can be categorized into three primary families :

A Priori Methods : These approaches demand that the decision-maker establish a compromise among the objectives before initiating the method. While a single execution may yield the desired solution, there is a potential for the outcome to be unsatisfactory.

Progressive Methods : This category necessitates the continuous involvement of the decision-maker throughout the problem-solving process to guide the search for solutions dynamically.

A Posteriori Methods : In contrast to the other two families, these methods do not consider the decision-maker's preferences during the solution search phase. It falls upon the designer to furnish the decision-maker with a well-distributed set of solutions, allowing them to choose the most suitable. Modeling user preferences in this context can be intricate, yet providing all possible solutions can sometimes be more costly.

Operations research methods predominantly employ a Posteriori methods for resolving multi-objective optimization problems, as proposed by Dhaenens-Flipo [37]. The objective is to identify a set of satisfactory solutions. However, this task is not always attainable, leading to a distinction between exact and approximate methods. Exact methods strive to precisely define the Pareto boundary of solutions. Notable examples include the method by evaluation and separation, along with the A^* algorithm, both of which will be detailed in the following section [38].

3.6 Exact Resolution Methods

A multitude of resolution methods exists in the literature for combinatorial optimization. Table 3.2 discusses different optimization methods, categorizing them into two distinct groups [39] :

- The first group comprises exact branching methods that ensure completeness in resolution. Examples include "Branch and Bound" and dynamic programming. The computational time required for resolution by such methods generally increases exponentially with the size of the problem to be solved (in the worst-case scenario).
- The second group includes approximate methods, aiming to find a high-quality solution within a reasonable computation time without guaranteeing optimality.
- Metaheuristics represent another significant part of approximate methods, offering intriguing avenues for designing heuristic methods in combinatorial optimization.

TABLE 3.1 – The resolution methods for multi-objective problems

Method Type	Description	Advantages and Disadvantages
A Priori Methods	Require a prior compromise from the decision-maker before execution.	— Simplicity and single execution. — Result may be unsatisfactory.
Progressive Methods	Decision-maker guides the solution search process.	— Decision-maker involvement throughout. — Requires continuous decision-maker presence.
A Posteriori Methods	Preferences not considered during search. Designer provides a range of solutions.	— Provides a diverse set of solutions. — Modeling user preferences can be challenging.
Weighted Sum	Combines objectives using weighted coefficients.	— Simplicity and ease of implementation. — Limited handling of non-convex Pareto fronts.
Pareto-based Methods	Identify Pareto-optimal solutions representing trade-offs.	— Captures the entire Pareto front. — Requires extensive computational resources.
Evolutionary Algorithms	Use evolutionary principles (e.g., genetic algorithms) for Pareto-optimal solutions.	— Handles non-linearity and non-convexity well. — Suitable for problems with many objectives. — Convergence may be slow.
Constraint Handling	Converts multi-objective problem into a single-objective problem with additional constraints.	— Compatible with existing single-objective solvers. — May result in loss of information about Pareto front.
Aggregated Methods	Transform multi-objective problem into a single-objective problem.	— Simplifies the problem into a single objective. — Loss of information about trade-offs.

TABLE 3.2 – Characteristics of the different resolution methods

Method Group	Characteristics
Exact Methods	— Guarantees optimality. — Examples : "Branch and Bound," dynamic programming. — Computational time may increase exponentially.
Approximate Methods	— Seeks good-quality solutions in a reasonable time. — Includes metaheuristics as innovative approaches. — Does not guarantee optimality.

The above discussion highlights two major groups of optimization methods : exact methods, which guarantee optimal solutions but may suffer from computational complexity, and approximate methods, which aim for good-quality solutions within acceptable time frames. The mention of "Branch and Bound" and dynamic programming exemplifies well-established exact methods, while the acknowledgment of metaheuristics underscores the significance of innovative approaches in combinatorial optimization. The detailed characteristics of resolution methods are discussed in Table 3.2.

3.6.1 The Evaluation and Separation Method

The evaluation and separation method plays a crucial role in combinatorial optimization, particularly in the context of exact algorithms such as "Branch and Bound." This method operates within the framework of a solution tree, where each node represents a potential solution to the optimization problem. The separation step involves dividing the set of feasible solutions contained within a node based on a specific criterion. It is essential to ensure that this division does not lead to any loss or addition of solutions. Specifically, the union of the subsets associated with the child nodes must precisely match the set associated with the parent node.

To optimize the algorithm's efficiency, the evaluation and bound components are incorporated. These components help decide whether to explore or prune certain nodes in the tree structure. If the evaluation of a node is less than or equal to the lower bound, or if the evaluation is exact, or if the node is infeasible, there is no need to further expand that node. This strategic pruning prevents unnecessary exploration, significantly reducing the computational effort required for exhaustive search.

The development strategy, influenced by the tree traversal approach, adds depth to the understanding of the problem-solving process. Whether opting for depth-first exploration until a prunable node is found, pursuing the most promising direction with best-first exploration, or adopting breadth-first expansion, these strategies shape the algorithm's behavior and contribute to the efficiency of combinatorial optimization in exact methods. the steps are explained below :

Separation : We divide, based on a certain criterion, the set of feasible solutions contained in a node of the tree structure. This separation should neither lose nor add solutions, meaning the union of the subsets associated with the children of a node must be equal to the set associated with that node.

Let S be the set of feasible solutions in a node, and let $S_1, S_2, \ldots, S_k$ be subsets associated with its children.

$$S = S_1 \cup S_2 \cup \ldots \cup S_k \tag{3.11}$$

Evaluation and Bound : These components prevent the need to split certain nodes in the tree structure. It is unnecessary to expand a node s in the following cases :

— The evaluation of node s is less than or equal to the lower bound.

— The evaluation of node s is exact.

— The node is infeasible.

Development Strategy : The strategy for node expansion directly depends on the traversal of the solution tree, and three main approaches are distinguished :

— **Depth-First** : A node is expanded in-depth until finding a node that can be pruned.

— **Best-First :** Explore the direction that seems most "promising."

— **Breadth-First.**

3.6.2 Dynamic Programming

Dynamic Programming (DP) is a powerful optimization technique used to solve problems that exhibit overlapping subproblems and optimal substructure. It is particularly effective in solving optimization problems, where the goal is to find the best solution from a set of feasible solutions. DP involves breaking down a problem into smaller, overlapping subproblems and solving each subproblem only once, storing the solutions to subproblems in a table (usually implemented using arrays or memoization) to avoid redundant computations. The stored solutions are then used to build up to the solution of the original problem [40]. Here are the key components and steps involved in dynamic programming :

— The problem has an optimal substructure if an optimal solution to the overall problem can be constructed from optimal solutions of its subproblems. This property allows breaking down a larger problem into smaller, more manageable subproblems.

— Subproblems recur many times, and the solutions to the same subproblems are reused rather than recomputed. This overlapping nature is exploited to optimize the algorithm by storing and retrieving intermediate results.

— This is a top-down approach where the solutions to subproblems are stored in a table, and before solving a subproblem, the algorithm checks whether the solution is already available in the table. If yes, it is retrieved ; if not, the subproblem is solved and its solution is stored.

— In this approach, the DP table is filled in a bottom-up manner, starting from the simplest subproblems and progressively solving more complex ones until the solution to the original problem is obtained.

TABLE 3.3 – The different types of heuristics and their characteristics

Heuristic	**Characteristics**
Greedy Heuristics	Make locally optimal choices at each stage with the hope of finding a global optimum.
Randomized Heuristics	Introduce randomness to explore the solution space and avoid getting stuck in local optima.
Metaheuristics	Higher-level strategies that guide the exploration of the solution space. Examples include genetic algorithms, simulated annealing, and ant colony optimization.
Constructive Heuristics	Build a solution piece by piece, iteratively improving it at each step.
Local Search Heuristics	Start with an initial solution and iteratively move towards better solutions in the neighborhood.

— The relationship between the solutions of subproblems and the overall problem is expressed in terms of state transitions. The DP transition equation defines how the solution to a larger problem can be built from the solutions to its subproblems.

Dynamic Programming is widely applied in various domains, including optimization problems, sequence alignment, shortest path problems, and many others. Its effectiveness lies in avoiding redundant computations, leading to significant improvements in time and space complexity compared to naive recursive approaches.

3.7 Heuristics

Heuristics are problem-solving strategies or rules of thumb that people use to quickly find satisfactory solutions to complex problems. Unlike optimal algorithms, heuristics do not guarantee the best solution but rather provide a practical and efficient approach to problem-solving. They are particularly useful in situations where finding an optimal solution is computationally expensive or impractical [41]. Below are some key characteristics and types of heuristics [42] :

— Heuristics prioritize efficiency and speed over optimality. They are designed to quickly reach a reasonably good solution without exhaustively exploring all possible alternatives.

— Heuristics are often rule-based strategies that guide decision-making. These rules are derived from experience, intuition, or domain-specific knowledge.

— Heuristics aim to provide approximate solutions that are "good enough" for the given problem. They might not guarantee the best outcome but are usually acceptable in real-world scenarios.

— Heuristics are typically simple and easy to apply. They rely on straightforward decision rules that do not involve complex computations.

— Heuristics are commonly used in optimization problems, decision-making, problem-solving, and artificial intelligence. Examples include greedy algorithms and hill climbing.

3.7.1 Example of Heuristics

Among the most well-known heuristics, we distinguish greedy algorithms and the A^* algorithm. We define these two approaches in the following subsections.

Greedy Algorithm

A Greedy Algorithm is an approach that selects, step by step, a local optimum Slavık [43]. In certain cases, this approach proves effective in reaching a global optimum. This method is applied in many combinatorial optimization problems, and its results are often satisfactory. While there is no guarantee of optimality, the solution's quality remains appreciable and is obtained in a relatively short time. Problems such as searching in graphs (minimum weight spanning tree), coin change algorithms, and graph coloring problems have been successfully addressed using this method. The key characteristic of greedy algorithms is that they make decisions based solely on the information available at the current step without considering the overall problem [44]. Here are the main components and characteristics of the Greedy Algorithm :

Components :

— **Greedy Choice Property :** At each step, the algorithm makes the best possible decision based on the current information without considering the consequences of this decision on future steps. This property is also known as making a locally optimal choice.

— **Optimal Substructure :** A problem has an optimal substructure if an optimal solution to the overall problem can be constructed from optimal solutions of its subproblems. The Greedy Algorithm relies on the assumption that choosing the locally optimal solution at each step leads to a globally optimal solution.

Characteristics :

— **Simplicity :** Greedy algorithms are typically simple and easy to understand. The decision-making process at each step is straightforward.

— **Efficiency :** Greedy algorithms are often efficient in terms of time complexity since they make decisions based on the current information without the need for extensive computations.

— **No Backtracking :** Once a decision is made, the algorithm does not revisit or change its choices. This lack of backtracking distinguishes greedy algorithms from some other optimization approaches.

— **Not Always Optimal :** While greedy algorithms are easy to implement and computationally efficient, they do not always guarantee an optimal solution. In some cases, the solution obtained may be suboptimal.

A^* Algorithm

The A^* algorithm is another notable heuristic approach used for problem-solving. This algorithm employs a systematic and iterative search strategy to explore solution spaces. It has been applied to various domains, including artificial intelligence and optimization problems [45].

These heuristic methods provide practical and efficient solutions, albeit without a guarantee of optimality. The Greedy Algorithm, in particular, is widely employed due to its simplicity and effectiveness in various combinatorial optimization scenarios. While it may not always yield the optimal solution, its ability to quickly find a satisfactory solution makes it a valuable tool in problem-solving.

3.8 Metaheuristics

Metaheuristics emerged in the late seventies with a common goal : to effectively solve challenging optimization problems. These methods are designed to address various types of single-objective or multi-objective optimization problems, constituting a subdomain of stochastic optimization. The term "stochastic" directly refers to the use of randomness, which serves as a double-edged sword for metaheuristics. Employing randomness is a necessary practice when the solution search space is vast. However, excessive use of randomness may lead us away from optimal solutions. All metaheuristics leverage this stochastic nature and tuning parameters to stay focused on solutions. The main challenge in using metaheuristics lies in making judicious choices for these parameters. While there are numerous suitable metaheuristics documented in the literature for various problem scenarios, selecting a specific method should be done with caution.

The objective of metaheuristics aligns with that of heuristics : to obtain high-quality solutions within a reasonable time. However, unlike heuristics, the algorithm and application of a metaheuristic are entirely independent of the specific problem being addressed. The key characteristics of metaheuristics is discussed below :

Characteristics

- **Generalization :** Metaheuristics are generic and can be adapted to various optimization problems without significant modification.
- **Exploration and Exploitation :** They strike a balance between exploring the solution space to discover new regions and exploiting known promising areas to improve solutions.
- **Iterative Improvement :** Metaheuristics iteratively refine candidate solutions, moving toward better solutions over time.
- **Stochastic Nature :** Many metaheuristics incorporate randomness or probabilistic components to diversify the search and avoid getting stuck in local optima.

Types of Metaheuristics

- **Simulated Annealing :** Inspired by the annealing process in metallurgy, this algorithm introduces randomness in the search process, allowing it to escape local optima.
- **Genetic Algorithms :** Mimicking the process of natural selection and genetics, genetic algorithms involve the evolution of a population of candidate solutions through selection, crossover, and mutation.
- **Ant Colony Optimization :** Based on the foraging behavior of ants, this algorithm models the interaction of artificial ants to find optimal paths in a solution space.
- **Particle Swarm Optimization :** Inspired by the social behavior of birds or fish, particle swarm optimization involves a population of particles that move through the solution space, adjusting their positions based on their individual and collective experiences.
- **Tabu Search :** This method uses memory structures to avoid revisiting previously explored solutions, preventing the algorithm from getting stuck in cycles.
- **Variable Neighborhood Search :** This approach systematically explores different neighborhoods of a solution, adjusting the search space dynamically.

Application Areas

- **Combinatorial Optimization :** Traveling Salesman Problem, Job Scheduling, Graph Coloring.
- **Numeric Optimization :** Function optimization, parameter tuning, and machine learning.
- **Engineering :** Network design, resource allocation, and process optimization.
- **Logistics and Transportation :** Routing, vehicle scheduling, and supply chain optimization.

Metaheuristics are high-level strategies or frameworks designed to guide and improve the performance of optimization algorithms. Unlike specific algorithms that are tailored to solve a particular problem, metaheuristics provide a more general approach that can be applied to a wide range of problems. These methods are often used when traditional optimization techniques become impractical or ineffective, especially in complex problems where finding an optimal solution is challenging.

3.8.1 Single Solution Metaheuristics

Single solution metaheuristics are optimization algorithms that operate on a single solution at a time. Unlike population-based metaheuristics, which maintain and evolve a population of solutions, single-solution metaheuristics focus on exploring and improving a single candidate solution iteratively. These methods are particularly suitable for solving optimization problems where maintaining and managing a population of solutions may be computationally expensive or impractical.

Simulated Annealing (SA)

Simulated Annealing is a probabilistic optimization algorithm inspired by the annealing process in metallurgy. It explores the solution space by allowing movements to less favorable solutions, analogous to the thermal annealing process. SA accepts worse solutions with a certain probability to escape local optima as given below in Algorithm 1.

Algorithm 1 Simulated Annealing

```
Initialize solution s and temperature T
while T > stopping criterion do
    Generate a neighboring solution s′
    Calculate the change in objective function : ΔE = f(s′) − f(s)
    if ΔE < 0 or rand(0,1) < e^(−ΔE/T) then
        Accept s′
    end if
    Update temperature T
end while
```

Tabu Search (TS)

Tabu Search is a local search algorithm that systematically explores the neighborhood of a current solution. It uses memory structures (tabu lists) to keep track of previously visited solutions and avoid revisiting them as given below in Algorithm 2. TS aims to guide the search toward promising areas [46].

Algorithm 2 Tabu Search

Initialize solution s, set tabu list TL to empty
while stopping criterion not met **do**
    Find the best neighboring solution s' not in TL
    Update tabu list TL
    Move to solution s'
end while

Variable Neighborhood Search (VNS)

Variable Neighborhood Search is an iterative optimization algorithm that systematically explores different neighborhoods of a solution. It dynamically adjusts the search space, moving between neighborhoods with varying levels of granularity to escape local optima as given below in Algorithm 3 [47].

Algorithm 3 Variable Neighborhood Search (VNS)

Initialize solution s, set initial neighborhood k
while stopping criterion not met **do**
    Generate a neighboring solution s' in neighborhood k
    if $f(s') < f(s)$ **then**
        Move to s'
        Set k to its initial value
    else
        Increment k
    end if
end while

Genetic Algorithms (GA)

Genetic Algorithms typically operate with populations, but a variant involves maintaining a single solution. This variant uses genetic operators such as crossover and mutation on a single candidate solution to explore the solution space as given below in Algorithm 4 [48].

Algorithm 4 Genetic Algorithm with Single Solution

Initialize solution s
while stopping criterion not met **do**
    Apply crossover and mutation operators to s
    Evaluate the offspring solutions
    Select the best offspring as the new solution s
end while

Iterated Local Search (ILS)

Iterated Local Search is a metaheuristic that iteratively applies a local search procedure to the current solution and perturbs the solution to escape local optima. It combines intensification through local search with diversification through perturbation as given below in Algorithm 5 [49].

Algorithm 5 Iterated Local Search (ILS)

```
1: Initialize solution s
2: while stopping criterion not met do
3:     Apply a local search procedure to improve s
4:     Perturb s to escape local optima
5: end while
```

3.8.2 Metaheuristic with Population of Solutions

Metaheuristics with a population of solutions represent a class of optimization algorithms that maintain and evolve a group of potential solutions to search for optimal or near-optimal solutions. Unlike single-solution metaheuristics, which work with only one solution at a time, these algorithms operate on a diverse set of solutions concurrently. The idea is to explore multiple regions of the solution space simultaneously, promoting diversity and avoiding premature convergence to suboptimal solutions [49].

Key Characteristics :

— **Population Representation :** A population comprises multiple individuals, each representing a potential solution to the optimization problem. Each individual in the population is a candidate solution with specific characteristics or parameters.

— **Parallel Processing :** The algorithm evaluates, selects, and evolves multiple solutions in parallel during each iteration or generation.

— **Diversity Maintenance :** Strategies are employed to maintain diversity within the population, preventing convergence to local optima. Techniques like recombination, mutation, or local search help explore different regions of the solution space.

Common Metaheuristics with Population

— **Genetic Algorithms (GAs)** : maintain a population of candidate solutions, allowing for exploration through crossover and mutation operations. Selection mechanisms determine which individuals contribute to the next generation [48].

— **Particle Swarm Optimization (PSO) :** maintains a swarm of particles, each representing a potential solution. Particles move through the solution space based on their own experience and the collective knowledge of the swarm.

— **Ant Colony Optimization (ACO) :** uses a population of artificial ants, each constructing solutions by depositing pheromones on solution components. The population collectively influences the exploration of the solution space.

The comparative analysis of common Metaheuristic methods are discussed below in Table 3.4.

Metaheuristics Workflow

— **Initialization :** Generate an initial population of solutions randomly or through heuristics. Evaluate the fitness of each solution in the population.

TABLE 3.4 – Comparison of the metaheuristics with a population of solutions.

Metaheuristic	**Genetic Algorithms (GAs)**	**Particle Swarm Optimization (PSO)**	**Ant Colony Optimization (ACO)**
Population Representation	Group of candidate solutions (chromosomes)	Swarm of particles	Population of artificial ants
Parallel Processing	Yes	Yes	Yes
Diversity Maintenance	Crossover and mutation operations	Particle movement based on local and global	Pheromone communication and exploration
Workflow Steps	Initialization, Selection, Crossover, Mutation, Evaluation, Next Generation	Initialization, Particle Movement, Evaluation, Update Best, Update Velocity	Initialization, Solution Construction, Pheromone Update, Solution Evaluation, Solution Update
Advantages	- Global exploration - High parallelism - Solution representation flexibility - Adaptive to various problem domains	- Simple implementation - Ease of implementation - Quick convergence in some cases	- Distributed problem-solving approach - Ability to handle combinatorial problems
Challenges	- Proper parameter tuning required - Computational overhead - Convergence may be slow	- Sensitivity to parameter settings - Limited exploration in complex spaces	- Sensitive to parameter settings - May converge prematurely in some cases - Complex parameter tuning
Applications	- Combinatorial optimization - Function optimization - Scheduling problems - Machine learning	- Function optimization, Machine learning	- Routing problems, TSP, Vehicle routing, scheduling, Network design

— **Iteration :** Conduct iterative generations where solutions evolve overtime. Apply selection mechanisms to choose individuals for reproduction.

— **Reproduction :** Use genetic operators (crossover, mutation) or swarm dynamics (particle movement) to create offspring solutions.

— **Evaluation :** Evaluate the fitness of the newly generated solutions.

— **Selection for Next Generation :** Choose individuals, based on fitness, for the next generation. Maintain diversity to explore different areas of the solution space.

— **Termination :** Check termination criteria (e.g., a predefined number of iterations or achieving a satisfactory solution).

— **Repeat :** If termination criteria are not met, repeat the iteration process.

The algorithm for the above workflow is given below in Algorithm 6 :

Algorithm 6 Genetic Algorithm with Population

```
Initialize a population of solutions P
while stopping criterion not met do
    Evaluate the fitness of each solution in P
    Select parents for crossover based on fitness
    Apply crossover to produce offspring
    Apply mutation to the offspring
    Evaluate the fitness of the offspring
    Select individuals for the next generation
    Replace the current population with the new generation
end while
```

Advantages :

— **Parallel Exploration :** The use of a population allows simultaneous exploration of multiple regions of the solution space.

— **Diversity :** Maintaining a diverse set of solutions reduces the risk of premature convergence to suboptimal solutions.

— **Global Exploration :** Population-based metaheuristics are well-suited for global optimization problems with complex solution spaces.

Challenges :

— **Computational Overhead :** Working with a population requires more computational resources than single-solution methods.

— **Parameter Tuning :** Proper tuning of parameters, such as population size and genetic operators, is critical for performance.

3.9 Hybrid Metaheuristics

Hybrid metaheuristics combine two or more different metaheuristic algorithms to create a more robust and efficient optimization approach. This integration aims to leverage the strengths of individual algorithms and mitigate their weaknesses [50]. The common hybridization metaheuristics are discussed below :

Sequential Hybridization : In sequential hybridization, different metaheuristics are applied one after another, with the output of one metaheuristic becoming the input for the next. This approach facilitates a sequential refinement of solutions, potentially leveraging the strengths of each metaheuristic in succession. An example of this is applying Genetic Algorithms (GAs) followed by Simulated Annealing (SA), where GAs might explore a wide search space initially, and SA performs local refinement on the solutions generated by GAs.

Parallel Hybridization : Parallel hybridization involves running multiple metaheuristics concurrently, with their solutions combined and evaluated to choose the best one. This approach allows for simultaneous exploration of multiple regions of the solution space. For instance, executing Genetic Algorithms and Particle Swarm Optimization concurrently enables the exploitation of different search strategies simultaneously.

Model-Based Hybridization : In model-based hybridization, a mathematical model represents the problem, and metaheuristics operate on this model. This approach abstracts the problem into a mathematical form that facilitates the application of various metaheuristics. An example includes coupling Genetic Algorithms with mathematical programming, where the mathematical model guides the search process.

Memory-Based Hybridization : Memory-based hybridization involves storing solutions obtained by one metaheuristic in memory, which another metaheuristic then utilizes to guide its search. This approach leverages past solutions to influence the exploration of the solution space. For instance, combining Genetic Algorithms with Tabu Search involves using the memory of previously visited solutions by Tabu Search to guide the exploration conducted by Genetic Algorithms.

Parameter Tuning Hybridization : Parameter-tuning hybridization dynamically adjusts the parameters of one metaheuristic based on the performance of another, aiming to improve overall performance. This adaptive approach allows metaheuristics to fine-tune their behavior based on feedback from other algorithms. For example, Adaptive tuning of Genetic Algorithms using Differential Evolution dynamically adjusts GA parameters based on the performance observed through Differential Evolution.

Crossover Hybridization : Crossover hybridization involves applying crossover operations from one metaheuristic within another, allowing for the combining of genetic information. This approach enables the exchange of genetic material between different solution representations. For example, applying Genetic Algorithms crossover within Particle Swarm Optimization facilitates the exchange of solution components between particles, enhancing diversity and exploration.

TABLE 3.5 – The different types of hybridization in metaheuristics

Hybridization	Description	Advantages	Examples
Sequential	Metaheuristics applied one after another	Exploits strengths of each	GA + SA
Parallel	Concurrent operation of multiple metaheuristics	Diverse exploration	GA & PSO
Model-Based	Metaheuristics operate on a mathematical model	Enhanced problem representation	GA + Mathematical Programming
Memory-Based	Solutions stored in memory guide another metaheuristic	Improved exploration	GA + Tabu Search
Parameter-Tuning	Dynamic adjustment of parameters based on performance	Improved adaptability	GA + Differential Evolution
Crossover	Crossover operations applied within another metaheuristic	Combined genetic information	GA Crossover within PSO
Solution Representation	Different metaheuristics operate on varied representations	Enhanced solution space coverage	GA (binary) + SA (continuous)

Solution Representation Hybridization : In solution representation hybridization, different metaheuristics operate on different representations of solutions, which are then combined to form a hybrid approach. This approach allows for leveraging the strengths of different solution representations. For instance, Genetic Algorithms operating on binary strings combined with Simulated Annealing on continuous representations enable the exploration of both discrete and continuous solution spaces simultaneously, expanding the search capabilities.

The comparative analysis is discussed below in Table 3.5 :

3.10 Summary

In this comprehensive exploration of Multi-Objective Optimization and Pareto Multi-Objective Optimization (PMO) problems, the present chapter has systematically addressed the challenges posed by simultaneous optimization of conflicting objectives. We categorized resolution methods into three distinct families : a priori, progressive, and a posteriori. A priori methods require predefined compromises, progressive methods involve continuous decision-maker guidance, and posterior methods furnish decision-makers with a set of solutions for informed choices. The discussion underscored the significance of exact and approximate techniques in operations research for resolving multi-objective optimization problems, emphasizing the balance between exploration and exploitation.

Furthermore, the chapter introduced heuristic and metaheuristic approaches, such as greedy algorithms, genetic algorithms, and variable neighborhood search, enriching the toolkit for tackling PMO. The taxonomy of resolution methods, coupled with a detailed comparison of characteristics, elucidates the diverse landscape of techniques available for PMO problems. The discussion on hybrid metaheuristics highlighted the potential advantages of amalgamating different methods to enhance overall performance. The chapter contributes valuable insights to the field by unraveling the complexities of multi-objective optimization, offering a nuanced understanding of resolution methods and their practical applicability.

Chapitre 4

State of the Art in WSNs Deployment

4.1 Introduction

WSNs deployed for surveillance applications have undergone significant advancements, with a focus on optimizing various aspects of deployment. SNs and RNs placement strategies vary from random deployment to deterministic strategies, with a growing emphasis on optimizing cost, hop-count, coverage, and connectivity. Ensuring comprehensive target point coverage is essential for effective surveillance, and recent research explores techniques to minimize the number of sensor nodes while maximizing coverage. Connectivity considerations are equally crucial, with algorithms designed to maintain network connectivity for reliable data transmission [51].

Multi-objective optimization has gained prominence, allowing simultaneous optimization of objectives like minimizing the number of SNs number of RNs and network diameter. These approaches utilize sophisticated algorithms to find optimal trade-offs, contributing to more efficient and sustainable surveillance applications [52]. Real-world deployments of WSNs for surveillance have intensified, particularly in smart city initiatives. These deployments tackle challenges unique to urban environments, including interference, scalability, and adaptability to dynamic conditions. Furthermore, the integration of WSNs with other technologies, such as cameras, drones, and machine learning, enhances surveillance capabilities. Machine learning algorithms enable advanced data analytics, including object recognition, anomaly detection, and pattern recognition, contributing to more intelligent and responsive surveillance systems. Overall, the state of the art in the WSN deployment for surveillance applications reflects a multidimensional approach, combining strategic placement, cost optimization, and advanced technologies to address the complex requirements of modern surveillance scenarios.

A WSN typically consists of numerous low-cost, low-power sensor nodes (SNs) capable of sensing, computation, and wireless communication. Despite their advantages, these networks face significant challenges, particularly related to energy consumption, network connectivity, and data accuracy, which directly impact their operational lifespan and effectiveness [53].

In the context of structural health monitoring, optimal wireless sensor placement (OWSP) is pivotal. The placement of sensors and sinks must minimize energy consumption while maximizing information effectiveness and balancing network connectivity and reliability. This problem has been approached using a dual-population constrained multi-objective optimization (DCCMO) algorithm, which has shown superior performance in terms of diversity and convergence compared to other state-of-the-art methods[54, 55].

Agricultural applications benefit significantly from the use of linear programming (LP) models, which help optimize resource utilization such as water, labor, and fertilizers. By maximizing profit through efficient resource allocation, these models play a vital role in improving agricultural productivity. Similarly, the deployment of real-time monitoring systems in water supply networks can help detect accidental pollution. Methods that optimize the number and location of sensors based on pressure differences and hierarchical clustering have been proposed to ensure maximum information capture with minimal sensors[56, 57].

Underground WSNs (WUSNs) present unique challenges due to highly lossy underground channels and the difficulty in recharging buried SNs. One solution involves deploying aboveground relay nodes (RNs) to relay traffic, thereby extending network lifetimes. The RN placement problem in WUSNs, which must consider factors such as load balancing and signal attenuation, has been tackled using a two-phase optimization method that shows promising results in maximizing network lifespan [58, 59].

In traditional WSNs, the use of powerful relay nodes to facilitate communication between sensor nodes can significantly enhance energy efficiency. Research has introduced polynomial-time approximation algorithms to minimize the number of relay nodes needed while ensuring connectivity, either through direct relay-sensor paths or exclusively relay-node paths [60, 61].

While WSNs hold immense potential across various domains, optimizing node placement, energy consumption, and network connectivity remains a complex challenge. The development of innovative algorithms and optimization techniques, as discussed in the referenced studies, is crucial for advancing WSN technology and its applications. This chapter consolidates these advancements, providing a comprehensive review of methodologies aimed at enhancing the efficiency, reliability, and lifespan of WSNs [62, 63].

WSN technologies reveal a profound impact on the advancement of smart environments across diverse fields, including manufacturing, smart cities, transport, health, and the Internet of Things (IoT) [64]. This chapter critically analyzes the current research trends to address the challenges associated with the WSN deployment optimization [65].

over the past decade, researchers have increasingly leveraged various methods to address specific challenges within WSNs. The chapter provides a comprehensive review of recent studies, guiding readers to understand the up-to-date applications of optimization algorithm in addressing the challenges of WSNs deployment, contributing to the overall efficiency and effectiveness of WSNs.

4.2 Review of Optimization Techniques for WSNs Deployment

In the era of exponential growth in IoT technology, new applications, and online services have emerged, with smart car parks being a notable example that leverages WSNs at the core of IoT. This chapter addresses the state of the art in the optimization of WSNs deployment. Generally, such networks comprise SNs deployed for target coverage within the car park and RNs tasked with relaying alert messages from SNs to the sink node. Our study introduces a Multi-Objective Binary Integer Linear Programming (MOBILP) approach aimed at simultaneously minimizing the number of SNs, RNs, and the maximum distance from SNs to the sink node while ensuring coverage and connectivity. In addition, with the rapid expansion and development of sensor networks, different metaheuristics to solve the WSN deployment optimization problem have been developed. Both the methods and approaches are discussed in this chapter.

4.2.1 The Linear Programming Approach

The authors in [66] addressed the deployment strategies that consider multi-objectives such as cost (number of sensors), coverage, network connectivity and lifetime. To do so, they formulated two variants of a mixed-integer linear programing (MILP) model. As a first step, they solve the problem of monitoring target points using a minimum number of connected sensor nodes. Then, the above formulation is extended to another MILP called the lifetime-aware model, which considers the requirement that the WSN must remain fully operational for a prescribed time period T.

The authors in [67] proposed a Linear programing (LP) that considers k targets and a set of candidate positions for SNs. It aims to place an optimal number of sensor nodes while satisfying k-Coverage (each target is covered by at least k sensor nodes) and m-connectivity (each sensor node has at least m-neighbor nodes). The authors proposed a meta-heuristic algorithm using the Bio-geography-based optimization to solve the problem. They built a multi-objective function from the weighted sum of the three objective functions : maximize target coverage, maximize the connectivity of each sensor node and minimize the total number of sensor nodes.

In [61], the author addresses the Optimal Wireless Sensor Placement (OWSP) in structural health monitoring, aiming to minimize energy consumption and maximize information effectiveness. OWSP is formulated as a constrained multi-objective optimization problem using mixed-integer programming.

4.2.2 The Metaheuristic Approach

With the rapid expansion and development of sensor networks, different metaheuristics to solve the WSN deployment optimization problem have been developed, accounting for a variety of objectives and constraints [68]. The types of metaheuristics are diversified based on their features, as stated in the article [69]. In this study, we discuss swarm-based metaheuristics to tackle the WSN deployment optimization. The SNs and RNs placement can be categorized into three distinct classes, namely SN deployment, RN deployment, and sequential placement of SNs and RNs.

SNs Placement Optimization

Authors in [70] have worked on the problem of SNs placement optimization with multiple objectives, including the maximization of the target points coverage, maximization of connectivity between the SNs and the sink node, minimization of the number of SNs, and minimization of the shortest distance between SNs and the sink node. The problem was aggregated to a mono-objective problem using the WSM [48]. The authors have then proposed using a meta-heuristic called Grey-wolf optimization (GWO) to solve it.

In [53], introduces a novel approach combining Particle Swarm Optimization and Iterated Local Search with the Optimal Position Determination (OPD) algorithm to optimize node placement in target-based WSNs. The goal of the proposed work is to achieve full coverage of target points target points and complete connectivity with the number of SN and RN. The method is utilized to determine the minimal set of sensor nodes required for coverage, while the OPD algorithm helps identify optimal positions for relay nodes to ensure connectivity. The approach demonstrates significant improvements over canonical PSO, Differential Evolution (DE), and Genetic Algorithms (GAs) in terms of both coverage and connectivity efficiency.

In [54], an optimization strategy is presented for sensor placement in Wireless Sensor Networks (WSNs) that prioritizes even energy distribution over precise sensor localization. The approach aims to maximize network lifetime while maintaining connectivity, making it suitable for applications like border zone control, battlefield surveillance, and fire prevention. The strategy addresses two optimization problems, i.e., short-term and long-term monitoring. Through mathematical analysis, an optimal sensor placement formula is derived, and a computer-based model using the OpNet discrete event simulator validates the strategy's effectiveness. The results indicate that the proposed strategy outperforms others by promoting uniform energy consumption across the network.

The study in [71] has suggested a Biogeography-Based Optimization (BBO) meta-heuristic for the optimization of WSNs deployment with multiple objectives, namely minimizing the sensing interference of SNs, maximizing the target points coverage, and minimizing the number of SNs, under the connectivity constraint. As in [70], the problem was aggregated to a mono-objective problem using the WSM.

In [48], a GA-based deployment approach has been proposed to find a minimal number of SNs in a way that all the target points are K-covered and all SNs are connected. Initially, the authors have presented the linear programming formulation of the problem followed by an explanation of the GA they proposed, along with an appropriate representation of the chromosomes, the derivation of the fitness function, selection, crossover, and mutation operators. The problem was formulated as a weighted sum of three objective functions for minimizing the number of SNs, maximizing the target points coverage, and maximizing network connectivity, respectively.

in [72], authors proposed a stochastic algorithm for optimal sensor deployment called Weighted Salp Swarm Algorithm (WSSA) where is an improved variant of Salp Swarm Algorithm (SSA). The proposed variant is applied for optimal sensor deployment task. This approach is applied on probabilistic sensing model to maximize coverage and radio energy model to minimize energy consumption. This strategy is a trade-off between coverage and energy efficiency of the sensor network. It was observed that WSSA algorithm outperformed many other stochastic algorithms in optimizing coverage and energy efficiency of Wireless Sensor Network (WSN).

RNs Placement Optimization

The work in [51] has focused on the optimization of the RNP in order to ensure the connectivity of a given set of SNs deployed with the sink node, where the objective is to minimize the number of RNs. As this problem is NP-hard, the authors have used meta-heuristics to solve it. They have proposed three meta-heuristics, notably the Moth Flame Optimizer (MFO) algorithm, Interior Search Algorithm (ISA), and Bat Algorithm (BA), to find a near-optimal solution in polynomial times.

In [73], the aim was to solve the problem of RNP. Consequently, the authors have adopted a meta-heuristic referred to as the Firefly Algorithm Based RNs Placement to minimize the number of RNs while satisfying three constraints, namely the coverage of each SN by at least one RN, the connectivity between RNs and the sink, and the consumed energy.

In [78], authors are interested in the problem of relay node placement, where there objective is modeled as a bi-objective problem with the goal of minimizing the average intra-cluster distance where it is de ned as the average of sum of distances of all the SNs from their selected RNs and the second objective is average hop-count from all the RNs to the sink to improve the network lifetime. For solving this problem where is NP-hard authors propose a multi-objective decomposition-based moth flame optimization (MOMFO/D) algorithm, that decomposes the objectives into multiple single objective which are optimized simultaneously.

TABLE 4.1 – Summary of the Related Research Works on the WSN Deployment Optimization Problem.

Ref	Node type		Constraint(s)	Objective function(s)	Method(s)	Remarks
	SN	RN				
[70]	√		None	Maximize target points coverage, maximize connectivity, minimize shortest distance and minimize the number of SNs	Grey-wolf optimization	Use SNs only for both coverage and connectivity. Thus, the lifetime of SNs could be an issue due to limited battery
[71]	√		Network connectivity	Minimize sensing interference, maximize target points coverage and minimize the number of SNs	Biogeography-based optimization	Use SNs only for both coverage and connectivity. Thus, the lifetime of SNs could be an issue due to limited battery
[48]	√		None	Maximize target points coverage, maximize connectivity and minimize the number of SNs	Genetic algorithm	Use SNs only for both coverage and connectivity. Thus, the lifetime of SNs could be issue due to limited battery
[51]		√	Network connectivity	Minimize the number of RNs	Moth flame optimizer algorithm, Interior search algorithm and bat algorithm	Deal with RN placement based on a pre-deployed SN which limits the search space
[73]		√	Coverage of each SN by at least one RN, connectivity between RNs and sink and energy consumption	Minimize the number of RNs	Firefly algorithm	Deal with RN placement based on a pre-deployed SN which limits the search space
[74]	√	√	None	**First step :** Maximize target points coverage and minimize the number of SNs. **Second step :** Maximize the connectivity rate of the SNs, maximize the connectivity rate of the RNs and minimize the number of RNs	Particle swarm optimization with an iterated local search algorithm	Use sequential deployment of SNs and RNs that gives priority to the SNs
[75]	√	√	Target points coverage and network connectivity	Minimize the number of SNs, minimize the number of RNs	Greedy coverage algorithm, redundancy removal algorithm and genetic algorithm	Use sequential deployment of SNs and RNs that gives priority to the SNs
[76]	√	√	Target points coverage and network connectivity	Minimize the number of nodes, minimize outage probabilities of the links in the network	Exact algorithm and heuristics	The problem was formulated as a sum with equal weights of objectives that finds a single solution
[77]	√	√	Target points coverage and network connectivity	Minimize the number of SNs, minimize the number of RNs and minimize ND	Multi-objective linear programming	Running time is high for large problem instances

In [63], the paper addresses the challenge of enhancing the efficiency and longevity of wireless sensor networks (WSNs) through strategic relay node placement. Given the low energy efficiency of long-distance transmissions by sensor nodes, deploying a few more powerful relay nodes can significantly prolong network life while maintaining connectivity. The study explores two versions of the relay node placement problem : Deploying the minimum number of relay nodes to ensure connectivity paths between sensor nodes, which may include both sensor and relay nodes. Deploying the minimum number of relay nodes to ensure connectivity paths between sensor nodes that consist solely of relay nodes.

In [79], the relay node placement problem in WSN is addressed with the aim of positioning a limited number of available relay nodes to enhance network performance by improving delivery ratio, reducing end-to-end delay, and ensuring connectivity in partially disconnected areas. This is particularly relevant in scenarios involving network repairs or dynamic environments where network characteristics must adapt to changing conditions. The problem is formalized using a linear, mixed integer mathematical programming model that incorporates specific constraints and penalty components to closely mimic the wireless environment. The model provides solutions that determine both the optimal relay node locations and the best data routing paths. A comprehensive simulation evaluation demonstrates the effectiveness of this approach, showing improved network performance compared to a state-of-the-art dynamic routing protocol and a relay node placement heuristic.

SNs and RNs Sequential Placement Optimization

In [74], the authors have implemented particle swarm optimization with an iterated local search algorithm (PSO-ILS) to minimize the number of SNs first for target points coverage and then minimize the number of RNs based on the pre-deployed SNs in the second step for the connectivity requirements.

In [75], the study has proposed a sequential approach based on three algorithms. First, a greedy coverage algorithm consists of placing SNs to ensure target points coverage. Second, a redundancy removal algorithm has been used to remove unnecessary SNs provided by the first algorithm in order to reduce the number of deployed SNs. A third algorithm based on the GA was used to place a minimal number of RNs to generate a connected graph of minimal length that connects the deployed SNs and the sink node.

Discussions

Table 4.1 provides a summary of the related research works on the WSN deployment optimization problem. The table shows that the existing research on the WSN deployment typically focuses on the SNs placement, RNs placement, or both SNs and RNs placement using the sequential approach. Commonly, most real-world applications require the deployment of both SNs and RNs. Consequently, the sequential placement of SNs and RNs is the widely used approach, which consists, firstly, of finding the optimal positions to place SNs to cover all target points. Then the optimal positions to place RNs are found to maintain connectivity between selected SNs and the sink node. Although this approach is frequently utilized, it has numerous drawbacks :

- This can lead to sub-optimal solutions as the placement of sensor nodes does not consider the placement of relay nodes and vice versa.
- The optimization of sensor nodes may result in nodes being placed in locations that do not provide the best connectivity to the sink node. Consequently, the optimization of relay

nodes may require additional nodes to be placed, leading to increased deployment costs and ND.

Accordingly, in [76], the authors have proposed exact as well as heuristic methods to solve respectively small and large instances of simultaneous deployment of nodes while minimizing the number of nodes and the outage probabilities of the links in the network. Furthermore, the problem in [76], the number of SNs and number of RNs, are taken as a single objective, which is a sum of the number of nodes. Due to this, only one solution can be found despite having conflicting objectives, each of which has a more efficient solution when tackled on its own. Therefore, in order to explore more thoroughly the solution search space compared to the research work in [76], recently in [77], we have suggested a MOLP approach that finds all possible non-dominated solutions (Pareto front). The work in [77] has provided an optimal solution to the simultaneous deployment of SNs and RNs in order to minimize the number of SNs and RNs and the ND. A serious limitation of the work of this work is its high computational time when the problem instance increases. In order to overcome such limitations and based on the aforementioned related works, meta-heuristics are recommended as an intelligent algorithm to solve the problem of the optimization of the WSN deployment for target points coverage and network connectivity. Therefore, in the present study, we propose a greedy chaos WOA for the simultaneous SNs' and RNs' deployment optimization problem, referred to as a GCWOA, to achieve a good balance between solution quality and computational time.

4.3 Conclusions

Much research on the deployment optimization of WSNs for surveillance applications has been devoted to address the challenges of achieving optimal coverage, connectivity, and target tracking. Multi-objective optimization has emerged as one of the key approaches, allowing the simultaneous optimization of conflicting objectives, such as minimizing the number of nodes, maximizing coverage, and ensuring network connectivity. The integration of WSNs with technologies like cameras, drones, and machine learning has further enriched surveillance capabilities, enabling advanced data analytics and intelligent responses. The literature review highlights the diverse range of deployment strategies, including Linear Programming approaches, Sequential Node Placement strategies, and Metaheuristic algorithms like Grey Wolf Optimization, Biogeography-Based Optimization, Genetic Algorithms, and Particle Swarm Optimization with Iterated Local Search.

While the existing research works offer valuable insights, the existing deployment solutions exhibit certain limitations, such as sub-optimality in sequential placement and computational challenges in Linear Programming for large instances, underscoring the urgent need for alternative solutions. In response, the following chapter will introduce a novel approach, based on linear programming, designed to provide a more effective balance between solution quality and computational efficiency for the simultaneous deployment optimization of SNs and RNs in WSNs.

Chapitre 5

Contribution I : Multi-objective Optimization of WSNs Deployment using Linear Programming

5.1 Introduction

The integration of WSNs within IoT systems has intensified in recent years through a set of technologies designed for interconnecting physical devices with each other and with the Internet [80]. A notable feature of IoT systems is their composition of nodes with constrained resources, including processing, communication, and storage [81]. In this study, WSNs serve as the core infrastructure of such systems, consisting of SNs, RNs, and sink nodes. The primary role of SNs is to collect information from the surrounding environment and transmit it via RNs to the sink node.

As exemplified in smart city applications, a smart car park [14] stands out as a typical instance of how IoT technology can be broadly applied to provide various services to users in everyday living environments.

The smart car park assists drivers in swiftly locating a free space in the nearby parking lot. In various parking lots, the determination of the availability or unavailability of a car park spot relies on ground SNs [82]. This solution demands the installation of one SN at each car parking spot, which could be costly, especially in large parking lots. Motivated by this observation, a camera-based WSNs [83] has been explored to ascertain the state of car park spots. Such a system can also be utilized for services like surveillance. The parking lot, being a sensitive area, poses considerable risks to cars, as indicated by recent car fire incidents summarized in Table 5.1. These incidents have prompted the design of an effective surveillance system for monitoring targets (car park spots) to prevent potential losses, which can pose risks to human safety and impact the earnings of insurance companies [84].

In this chapter, the problem of deterministic WSNs deployment [85] is of interest. Numerous techniques have been developed to address this problem [86]. Typically, existing studies have concentrated on placing SNs [87] and subsequently RNs [88], aiming to reduce deployment costs while maintaining low network delay and energy consumption. Contrary to these techniques, we argue that the simultaneous placement of both SNs and RNs may represent a superior alternative. Through a series of tests, we demonstrate that simultaneous placement surpasses existing solutions reported in the literature [89].

The focus of this study is on the simultaneous placement of SNs and RNs using a multi-objective formulation. The objectives involve minimizing the number of SNs, the number of

TABLE 5.1 – Statistics on car park fire incidents

Fire Date	Car park Fire places	Number of damaged cars
2020	Norway airoport	+1000
2020	Southwest Florida International Airport	+3500
2019	Aero India Show	+300
2017	Liverpool car park	+1000
2016	Winchester, England	80

RNs, and the network diameter (representing the length of the longest path among all shortest paths between SNs and the sink node). These objectives are pursued under coverage and connectivity constraints. It is noteworthy that in our case, the minimization of the diameter is considered to enhance the end-to-end delay of alert delivery.

5.2 Network description and assumptions

The WSN is composed of two fundamental sets, namely a set of SN and a set of RN. The SNs have the role of monitoring the spots inside the parking lot to obtain information on the state of car park spots (free or occupied parking places), and whether there is a fire in the cars parked in the spots. The RNs are used to relay data packets (spots status or fire alarm) generated by the SNs, up to the sink node.

5.3 Problem Formulation

As shown figure 5.1, using a discretization of the area of interest (car park space), we obtain a set of points, denoted $\mathcal{T}$. Let $\mathcal{T} = \{T_1, T_2, T_3, ..., T_k\}$ be the set of k targets (car park spots) distributed inside the area of interest, which we want to cover by SNs. Each SN is characterized by a sensing range R_s, assumed to be constants. SNs and RNs can be placed only at specific candidate positions to place SN (CPSs), CPS= $\{sn_1, sn_2, sn_3, ..., sn_n\}$ and candidate position to place RN (CPRs), CPR= $\{sr_1, sr_2, sr_3, ..., sr_m\}$, respectively. SN sn_j covers the target T_i if the sensing range of sn_j completely covers the target T_i. In other words, if the euclidean distance between the SN sn_j and the target T_i is no more than R_s then the target T_i is assumed to be covered by sn_j. On the other hand, two nodes are neighbors and can communicate (send/receive information) if the Received Signal Strength Indicator (RSSI) value perceived by a node is greater than or equal to RX_{Tsh} [1] which is the sensitivity of the receiving antenna. The radio propagation model used in this chapter is Log-normal shadowing model [90].

1. Represents the minimum RSSI threshold for communication

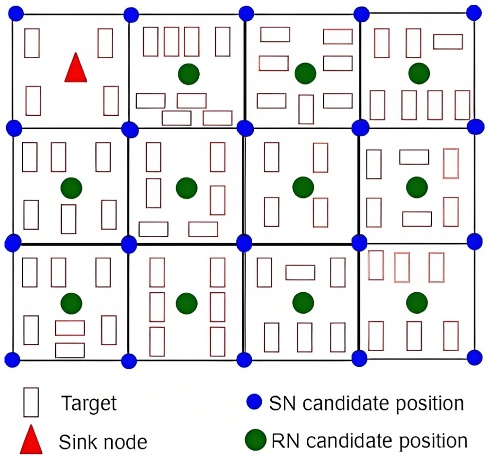


FIGURE 5.1 – Discretization of the deployment space in the 3×4 grid with 20 $cps \in$ CPS, 11 $cpr \in$ CPR, 1 sink node and 73 target points.

We consider the problem as a graph consisting of the union of sets of n CPSs, m CPRs, and one sink node V_1. Each SN i has a list s_i of target points that can monitor them. The objectives are to minimize :

— The number of deployed SNs, i.e., find $|V_s^{opt}|$.

— The number of deployed RNs, i.e., find $|V_c^{opt}|$.

— The Maximum distance of the shortest paths (hop-count) between all SNs to the sink node, i.e., $max_{i \in V_s^{opt}}\{dist_{i,V_1}\}$.

under the two following constraints :

— The union of elements associated with nodes of optimal SN set covers the set of target T. $\{s_i\} = T, \forall i \in V_s^{opt}$.

— Every SN not neighbor of the sink must be connected to the sink node directly or using at least one path composed only of RNs (Two-tiered architecture [91]).

5.4 The Optimization Problem

In this section, we first give some definitions. Then, we present the MOBILP, and describe the resolution method and how to deal with the objectives.

5.4.1 Definitions

Table 6.2 summarizes all of our model's related parameters and variables.

TABLE 5.2 – Parameters and variables

Parameter	**Definition**
Sink	is the index of the sink node
V_s	is the set of CPS.
V_r	is the set of CPR.
T	is the set of target points.
V^t_{cps}	Subset of V_s that can cover the target t.
V_i	Subset of V_r or the sink node which can communicate directly with node i ($i \in$ (CPS $\cup$ CPR))
V_1	Set of ($V_s \cup V_r$) that can communicate directly with the sink node.
H	Maximum number of hop-count allowed
Variables	
${x_i}^h$	if an SN is placed at CPS i, and situated at h-hop from the sink node.
${r_i}^h$	if a RN is placed at CPR i, and situated at h-hop from the sink node.
y_h	if there is a path with length h-hop from RN to the sink node.

5.4.2 Multi-objective Binary Integer Linear Program

Using the above parameters and decision variables, the linear model can be formulated as follows :

$$\min F1 = \sum_{i\in V_s} \sum_{h=1}^{H+1} x_i^h$$

$$\min F2 = \sum_{i\in V_r} \sum_{h=1}^{H} r_i^h$$

$$\min F3 = \sum_{h=1}^{H} y_h$$

Subject to :

$$\sum_{h=1}^{H+1} x_i^h \leq 1 \qquad i \in V_s \tag{5.1}$$

$$\sum_{h=1}^{H} r_i^h \leq 1 \qquad i \in V_r \tag{5.2}$$

$$\sum_{h=1}^{H+1} \sum_{i \in V_{cps}^t} x_i^h \geq 1 \qquad \forall t \in T \tag{5.3}$$

$$x_i^h \leq \sum_{j \in V_i} r_j^{h-1} \qquad \forall i \in V_s \backslash V_1, \forall h = 2 \dots H+1 \tag{5.4}$$

$$r_i^h \leq \sum_{j \in V_i} r_j^{h-1} \qquad \forall i \in V_r \backslash V_1, \quad \forall h = 2 \dots H \tag{5.5}$$

$$\sum_{h=2}^{H+1} x_i^h \leq 0 \qquad i \in V_1 \backslash V_r \tag{5.6}$$

$$\sum_{h=2}^{H} r_i^h \leq 0 \qquad i \in V_1 \backslash V_s \tag{5.7}$$

$$x_i^1 = 0 \qquad i \in V_s \backslash V_1 \tag{5.8}$$

$$r_i^1 = 0 \qquad i \in V_r \backslash V_1 \tag{5.9}$$

$$y_h \geq r_i^h \qquad i \in V_r, \quad h = 1 \dots H \tag{5.10}$$

$$y_h \leq \sum_{i \in V_r} r_i^h \qquad h = 1 \dots H \tag{5.11}$$

$$y_h \leq y_{h-1} \qquad h = 2 \dots H \tag{5.12}$$

$$y_h \leq \sum_{l=h+1}^{H+1} \sum_{i \in V_s} x_i^l \qquad h = 1 \dots H \tag{5.13}$$

The objective is to minimize simultaneously, F1 : number of SNs, F2 : number of RNs and F3 : diameter of the network.

The constraints (5.1) and (6.4) ensure that each SN and RN has only one path to the sink node. The constraint (6.5) ensures that each target point $t \in T$ is covered by at least one SN which is located at h hops from the sink node. The constraints (6.6) and (6.7) guarantees that each SN or RN located at h hops from the sink ($h > 1$), has at least one neighbor RN located at $h-1$ hops from the sink node. The constraints (6.8) and (6.9) guarantee that a SN or RN respectively, directly neighbor to the sink node cannot be situated at $h > 1$ hop. The constraints (6.10) and (6.11) avoid the inconsistency of the model, that is to say that no SN and no RN respectively is neighbor to the sink node unless it is located at 1-hop from it.

The constraint (5.10) determines the length of the paths made up of RNs towards the sink node. If the i^{th} RN located at h hops from the sink then there is a path of length k. The constraint (5.11) ensures that if there is a path of length h hops, it means that there is at least one RN located at h hops from the sink node. This is to avoid the inconsistency of the model. The constraint (5.12) guarantees that the existence of a path of length h hops induces the presence of a path of length $h-1$ hops. The constraint (5.13) guarantees that if there is a path with h hops length, a SN is necessarily selected at l hops from sink node ($l = (h+1) \vee l = (h+2) \vee \dots \vee l = (H+1)$).

5.4.3 Resolution Method

To produce the set of effective solutions, we choose the ε-constraint approach[92], which has the advantage of being extremely simple to implement. As shown in algorithm 1, this method consists in transforming the multi-objective problem to a mono-objective problem by considering one objective to optimize among the others and making the remaining objectives as bounded constraints.

We solve the linear program iteratively (line 4), where we minimize the objective function F1 while the objective functions F2 and F3 are considered as constraints delimited respectively by ε_2 and ε_3. For each iteration, we increase the bounds (ε_2 or ε_3) to obtain a new solution.

Algorithm 7 ε-constraint method

Input : $\underline{F_2}$ and $\overline{F_2}$ are respectively the lower and upper bounds of F_2, $\underline{F_3}$ is lower bound of F_3.
Output : P is the set of efficient solutions. $P \leftarrow \varnothing$

1: $\varepsilon_2 \leftarrow \underline{F_2}$
2: **while** $\varepsilon_2 \leq \overline{F2}$ **do**
3: $\quad$ $\varepsilon_3 \leftarrow \underline{F_3}$
4: $\quad$ **while** No improvement after six successive runs **do**
5: $\quad\quad$ $X \leftarrow Minimize(F_1, F_2 \leq \varepsilon_2, F_3 \leq \varepsilon_3)$
6: $\quad\quad$ **while** $\bigcirc X' \in P$ such that X' is better than X **do**
7: $\quad\quad\quad$ $P \leftarrow P \bigcup \{X\}$
8: $\quad\quad$ **end while**
9: $\quad\quad$ $\varepsilon_3 \leftarrow \varepsilon_3 + 1$
10: $\quad$ **end while**
11: $\quad$ $\varepsilon_2 \leftarrow \varepsilon_2 + 1$
12: **end while**
13: Return P

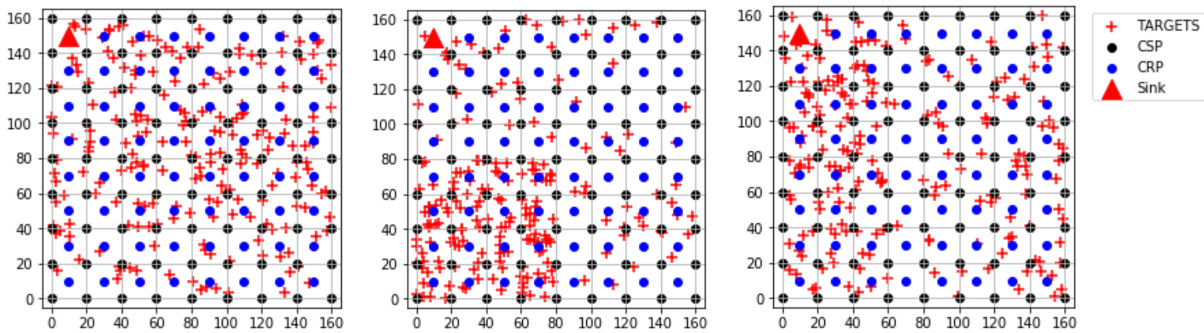
5.5 Experimental Findings

To evaluate the performance of the proposed MOBILP, we conducted extensive tests built on several scenarios. In this section, we describe the considered testbed, the experiments and discuss their results.

5.5.1 Testbed Setup

To simulate a test environment corresponding to a car park, we consider a square area of 160m x 160m divided into an equally-sized mesh grid of 20m x 20m. We define the V_s set as the vertices of the whole meshes, the V_r set as the center of each mesh, and the sink position is picked from the set of V_r. All nodes have the same Tx power of -5 dbm and radio sensitivity of -100 dbm, and all SNs have the same sensing range of 25 meters. Three scenarios are studied to show the effectiveness of our model MOBILP denoted *technique 3*, compared to research work in [93] and [94] using sequential deployment technique denoted *technique 2*, and research work in [95] using mono-objective simultaneously deployment technique denoted *technique 1*.

At this stage of research work, and in the absence of real data, we generate the following scenarios : **Scenario 1** (Figure 5.2a, Random distribution of all targets in the area of interest, **Scenario 2** (Figure 5.2b), Random distribution of 60% of targets in the left bottom quarter of the grid (near to the car park entrance in bottom left of the grid) and the remaining 40% is distributed randomly on the other three quarters, and **Scenario 3** (Figure 5.2c), Random distribution of 60% of targets in the three first columns of the grid and the remaining 40% is distributed randomly on the other columns. We vary the sink location for each scenario in the set {top left, center, bottom right}. For each sink location, the number of targets varies and takes the specified values of 50, 100, and 200 targets. Figure 5.2 depicts a few generated scenarios built from 200 targets and where the sink is located in the top left of the grid. The mathematical model is solved using "ε-Constraint algorithm" in which we call the solver Gurobi 9.0.3 [96], and executed on a PC with a two-core Intel Core Processor (2.50 GHz) and 6 GB RAM.



(A) (B) (C)

FIGURE 5.2 – The study scenarios : distribution of 200 targets with the sink in top left

5.5.2 Results and Discussions

We carried out several tests, the outcomes from program runs are depicted in tables 5.3, 5.4 and 5.5. In comparison to technique 1, that is the mono-objective optimization with simultaneous placement of SNs and RNs, MOBILP provides a set of alternative solutions which include the solution obtained from the technique 1. On the other hand, we observe that all the solutions provided by technique 2 are also included in the set of solutions provided by MOBILP or are dominated by those solutions. Therefore, modelling a simultaneous deployment problem as a multi-objective problem allows the decision-maker more flexibility to choose the optimal trade-off between conflicting objectives according to the context of the problem. Thus, the decision-maker can pick any specific number of RNs (#RN) and obtain the minimum associated number of SNs (#SN) and network diameter (ND).

There is an impact of target density as shown in Tables 5.3, 5.4 and 5.5. We note that when the density of targets increases inside the smart parking, more SNs and RNs are required, and hence the cost of the deployment increases significantly. This can be explained by the limited sensing range of the SNs which reduces the number of SNs that cover the same target and leads to the increase of the number of SNs and RNs to ensure coverage of all targets and connectivity requirements. Furthermore, the variation of the distribution of the target positions according to the three scenarios detailed above has an impact on the WSN deployment as shown in Tables 5.3, 5.4 and 5.5. Indeed, in scenario 2 when all the targets are close as they are concentrated in one area, the #SN has the lowest value. This is because one SN can cover more targets.

In addition, as illustrated in Tables 5.3, 5.4 and 5.5, it appears that the sink position affects only the diameter of the network whatever the scenario. We can notice that the best position of the sink to get the better value of the diameter is in the middle of the area under study, as

TABLE 5.3 – Test results : Sink at the top left

<table>
<tr><td></td><td colspan="3">Scenario 1</td><td colspan="3">Scenario 2</td><td colspan="3">Scenario 3</td></tr>
<tr><td></td><td>#SN</td><td>#RN</td><td>D</td><td>#SN</td><td>#RN</td><td>D</td><td>#SN</td><td>#RN</td><td>D</td></tr>
<tr><td></td><td colspan="9">50 targets</td></tr>
<tr><td>App1</td><td>18</td><td>12</td><td>7</td><td>13</td><td>10</td><td>6</td><td>19</td><td>11</td><td>7</td></tr>
<tr><td>App2</td><td>18</td><td>13</td><td>7</td><td>13</td><td>11</td><td>6</td><td>19</td><td>12</td><td>7</td></tr>
<tr><td rowspan="4">MOBILP</td><td>18</td><td>12</td><td>7</td><td>13</td><td>9</td><td>8</td><td rowspan="2">19</td><td rowspan="2">11</td><td rowspan="2">7</td></tr>
<tr><td rowspan="3">19</td><td rowspan="3">11</td><td rowspan="3">7</td><td rowspan="2">13</td><td rowspan="2">10</td><td rowspan="2">6</td></tr>
<tr><td rowspan="2">20</td><td rowspan="2">13</td><td rowspan="2">6</td></tr>
<tr><td>14</td><td>9</td><td>7</td></tr>
<tr><td></td><td colspan="9">100 targets</td></tr>
<tr><td>App1</td><td>23</td><td>13</td><td>7</td><td>21</td><td>11</td><td>7</td><td>22</td><td>12</td><td>7</td></tr>
<tr><td>App2</td><td>23</td><td>13</td><td>7</td><td>21</td><td>12</td><td>7</td><td>21</td><td>13</td><td>7</td></tr>
<tr><td rowspan="4">MOBILP</td><td>23</td><td>12</td><td>8</td><td rowspan="4">21</td><td rowspan="4">11</td><td rowspan="4">7</td><td rowspan="2">21</td><td rowspan="2">12</td><td rowspan="2">7</td></tr>
<tr><td>23</td><td>13</td><td>7</td></tr>
<tr><td>24</td><td>12</td><td>7</td><td rowspan="2">22</td><td rowspan="2">11</td><td rowspan="2">7</td></tr>
<tr><td>25</td><td>11</td><td>8</td></tr>
<tr><td></td><td colspan="9">200 targets</td></tr>
<tr><td>App1</td><td>33</td><td>11</td><td>7</td><td>26</td><td>13</td><td>6</td><td>30</td><td>14</td><td>7</td></tr>
<tr><td rowspan="2">App2</td><td>31</td><td>15</td><td>8</td><td>26</td><td>12</td><td>9</td><td>30</td><td>15</td><td>8</td></tr>
<tr><td>31</td><td>16</td><td>7</td><td>26</td><td>13</td><td>6</td><td>30</td><td>16</td><td>7</td></tr>
<tr><td rowspan="7">MOBILP</td><td>31</td><td>13</td><td>8</td><td>26</td><td>11</td><td>8</td><td>30</td><td>13</td><td>8</td></tr>
<tr><td>31</td><td>14</td><td>7</td><td>26</td><td>12</td><td>7</td><td>30</td><td>14</td><td>7</td></tr>
<tr><td>32</td><td>12</td><td>7</td><td>26</td><td>13</td><td>6</td><td>31</td><td>13</td><td>7</td></tr>
<tr><td rowspan="2">32</td><td rowspan="2">13</td><td rowspan="2">6</td><td>27</td><td>11</td><td>7</td><td>31</td><td>14</td><td>6</td></tr>
<tr><td rowspan="2">27</td><td rowspan="2">12</td><td rowspan="2">6</td><td rowspan="2">32</td><td rowspan="2">12</td><td rowspan="2">7</td></tr>
<tr><td rowspan="2">33</td><td rowspan="2">11</td><td rowspan="2">7</td></tr>
<tr><td>28</td><td>10</td><td>8</td><td>32</td><td>13</td><td>6</td></tr>
</table>

TABLE 5.4 – Test results : Sink at the centre

	Scenario 1			Scenario 2			Scenario 3		
	#SN	#RN	D	#SN	#RN	D	#SN	#RN	D
	50 targets								
App1	18	11	4	13	7	3	19	12	4
App2	18	13	4	13	8	3	19	12	6
							19	13	4
MOBILP	18	11	4	13	7	3	19	11	5
							19	12	4
							20	10	5
							20	11	4
	19	10	4				20	13	3
							21	10	4
							21	12	3
	100 targets								
App1	24	10	4	21	11	4	21	12	4
App2	23	12	4	21	12	5	21	13	4
				21	13	4			
MOBILP	23	11	4	21	11	4	21	12	4
							22	11	4
	24	10	4	22	10	5	24	10	4
	200 targets								
App1	32	12	3	26	11	3	31	12	3
App2	31	14	5	26	12	4	30	14	4
	31	15	4	26	13	3			
MOBILP	31	12	5	26	11	3	30	12	5
	31	13	4				30	13	4
	32	11	6				31	11	4
							31	12	3
	32	12	3	29	10	7	32	10	5
	33	11	4				32	11	3

TABLE 5.5 – Test results : Sink at the bottom right

<table>
<tr><td></td><td colspan="3">Scenario 1</td><td colspan="3">Scenario 2</td><td colspan="3">Scenario 3</td></tr>
<tr><td></td><td>#SN</td><td>#RN</td><td>D</td><td>#SN</td><td>#RN</td><td>D</td><td>#SN</td><td>#RN</td><td>D</td></tr>
<tr><td></td><td colspan="9">50 targets</td></tr>
<tr><td>App1</td><td>18</td><td>11</td><td>5</td><td>13</td><td>8</td><td>4</td><td>19</td><td>12</td><td>5</td></tr>
<tr><td rowspan="4">App2</td><td>18</td><td>13</td><td>7</td><td rowspan="3">13</td><td rowspan="3">8</td><td rowspan="3">6</td><td rowspan="2">19</td><td rowspan="2">12</td><td rowspan="2">6</td></tr>
<tr><td rowspan="2">18</td><td rowspan="2">14</td><td rowspan="2">6</td></tr>
<tr><td rowspan="2">19</td><td rowspan="2">13</td><td rowspan="2">5</td></tr>
<tr><td>18</td><td>15</td><td>5</td><td>13</td><td>9</td><td>4</td></tr>
<tr><td rowspan="3">MOBILP</td><td rowspan="2">18</td><td rowspan="2">11</td><td rowspan="2">5</td><td rowspan="2">13</td><td rowspan="2">8</td><td rowspan="2">4</td><td>19</td><td>12</td><td>5</td></tr>
<tr><td>20</td><td>11</td><td>5</td></tr>
<tr><td>19</td><td>10</td><td>7</td><td>15</td><td>7</td><td>5</td><td>21</td><td>10</td><td>8</td></tr>
<tr><td></td><td colspan="9">100 targets</td></tr>
<tr><td>App1</td><td>23</td><td>12</td><td>5</td><td>22</td><td>11</td><td>5</td><td>21</td><td>13</td><td>5</td></tr>
<tr><td rowspan="3">App2</td><td>23</td><td>11</td><td>9</td><td rowspan="2">21</td><td rowspan="2">11</td><td rowspan="2">6</td><td rowspan="2">21</td><td rowspan="2">12</td><td rowspan="2">7</td></tr>
<tr><td>23</td><td>12</td><td>6</td></tr>
<tr><td>23</td><td>14</td><td>5</td><td>21</td><td>12</td><td>5</td><td>21</td><td>13</td><td>5</td></tr>
<tr><td rowspan="6">MOBILP</td><td rowspan="2">23</td><td rowspan="2">11</td><td rowspan="2">7</td><td rowspan="2">21</td><td rowspan="2">11</td><td rowspan="2">6</td><td>21</td><td>12</td><td>6</td></tr>
<tr><td>21</td><td>13</td><td>5</td></tr>
<tr><td>23</td><td>12</td><td>5</td><td>21</td><td>12</td><td>5</td><td>22</td><td>11</td><td>6</td></tr>
<tr><td rowspan="2">24</td><td rowspan="2">11</td><td rowspan="2">5</td><td rowspan="2">22</td><td rowspan="2">10</td><td rowspan="2">7</td><td>23</td><td>12</td><td>5</td></tr>
<tr><td>25</td><td>10</td><td>6</td></tr>
<tr><td>25</td><td>10</td><td>6</td><td>22</td><td>11</td><td>5</td><td>25</td><td>11</td><td>5</td></tr>
<tr><td></td><td colspan="9">200 targets</td></tr>
<tr><td>App1</td><td>31</td><td>13</td><td>5</td><td>26</td><td>12</td><td>5</td><td>30</td><td>12</td><td>5</td></tr>
<tr><td rowspan="3">App2</td><td rowspan="2">31</td><td rowspan="2">13</td><td rowspan="2">8</td><td rowspan="3">26</td><td rowspan="3">13</td><td rowspan="3">5</td><td>30</td><td>14</td><td>7</td></tr>
<tr><td>30</td><td>15</td><td>6</td></tr>
<tr><td>31</td><td>14</td><td>5</td><td>30</td><td>16</td><td>5</td></tr>
<tr><td rowspan="8">MOBILP</td><td>31</td><td>12</td><td>8</td><td rowspan="3">26</td><td rowspan="3">12</td><td rowspan="3">5</td><td rowspan="2">30</td><td rowspan="2">12</td><td rowspan="2">5</td></tr>
<tr><td rowspan="2">31</td><td rowspan="2">13</td><td rowspan="2">5</td></tr>
<tr><td rowspan="3">31</td><td rowspan="3">11</td><td rowspan="3">6</td></tr>
<tr><td>31</td><td>11</td><td>8</td><td rowspan="3">27</td><td rowspan="3">11</td><td rowspan="3">5</td></tr>
<tr><td rowspan="3">32</td><td rowspan="3">12</td><td rowspan="3">5</td></tr>
<tr><td rowspan="3">32</td><td rowspan="3">11</td><td rowspan="3">5</td></tr>
<tr><td rowspan="2">28</td><td rowspan="2">10</td><td rowspan="2">5</td></tr>
<tr><td>33</td><td>11</td><td>6</td></tr>
</table>

shown in the Table 5.4, where all SNs are near the sink node. Finally, we can conclude that MOBILP generally outperforms the other techniques or at least provides the same solutions in all scenarios.

On the other hand, MOBILP takes longer running time compared to the other two techniques. For instance, In scenario 1, when the number of targets is 50 the techniques 1,2 and MOBILP solve the problem in 2.67s, 3.02s and 35.88s respectively. While in scenario 2 they solve the problem in 5.54s, 3.95s and 21.27s. Finally, for scenario 3 they take 2.20s, 2.29s and 15.32s respectively. This can be explained by the simultaneous deployment that increases the research space of the problem and the multi-objective version that makes the problem more complex. On the other side, the running time of the MOBILP increases when the number of targets increases. For example, to solve scenario 1 with 50, 100 and 200 targets the MOBILP takes 35.88s, 117.61s and 598.92s respectively. Nevertheless, we can agree on a reasonable resolution time as we are working offline. Indeed, we have considered the car park spots as the targets, not the vehicles.

5.6 Conclusions

This chapter has focused on the optimal deployment of WSNs in a smart car park. Our technique simultaneously identifies the locations of SNs and RNs, minimizing the number of SNs, RNs, and the network diameter while adhering to coverage and connectivity constraints. The problem was formulated as a graph, and we employed the "MOBILP" to address and solve it. Extensive testing was conducted across three scenarios : a purely random distribution and two scenarios representing realistic conditions with varying target distributions. The MOBILP exhibited superior performance and enhanced flexibility in achieving optimal compromises between conflicting objectives. In comparison to existing techniques—a sequential deployment of SNs followed by RNs and a simultaneous deployment with a mono-objective function yielding only one solution— the MOBILP consistently provided better solutions.

Chapitre 6

Contribution II : Simultaneous Sensor and Relay Nodes Deployment Optimization

6.1 Introduction

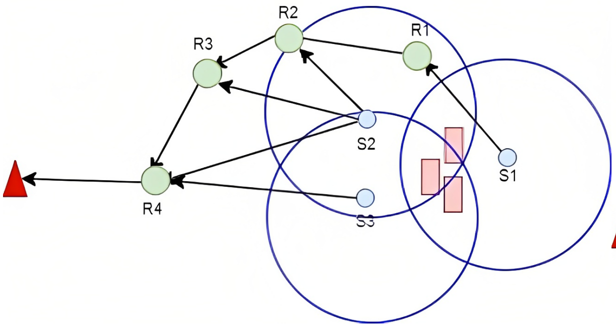
WSNs are increasingly being deployed in SCP applications as the IoT gains prominence [5]. These networks utilize SNs with limited computation and communication resources to monitor the environment and cover critical target points [97]. A sink node collects information from these SNs. Due to the limited communication range of SNs, especially due to low antenna height [98], multi-hop WSNs rely on RNs with higher communication ranges to establish network connectivity and forward data between RNs and the sink node.

However, deploying a fully connected WSN with many nodes for applications like SCP surveillance can be cost-prohibitive [99]. The expenses associated with node procurement, installation, and maintenance are significant. As a consequence, minimizing the number of deployed SNs and RNs can help reduce network deployment costs. The node deployment problem, also known as the node placement problem, has been proven to be NP-Hard in the literature.

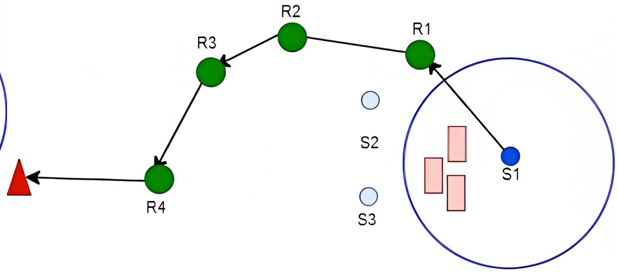
The deployment of WSNs involves two main aspects : Sensor Node Placement (SNP) and Relay Node Placement (RNP). The SNP problem focuses on identifying ideal locations to place the smallest number of SNs while meeting connectivity and coverage constraints [100]. On the other hand, the RNP problem entails placing the minimum number of RNs to connect all pre-deployed SNs with a sink node [101].

To illustrate the deployment strategy, Figure 6.1a presents three potential locations for SNs (denoted as CPS in light blue), four positions for RNs (denoted as CPR in light green), three target points (depicted as a red rectangle), and a sink node (represented by a red triangle). These target points might correspond to parking spots in a smart car park scenario. Using a sequential deployment technique, Figure 6.1b depicts the solution, where SN S1 is placed in the first step to cover all target points, followed by the deployment of RNs (R1, R2, R3, and R4) to connect SN S1 to the sink node. In contrast, the simultaneous deployment of SNs and RNs assigns equal priority to both, resulting in two "non-dominating" solutions shown in Figure 1b and Figure 6.1c, where neither is superior to the other.

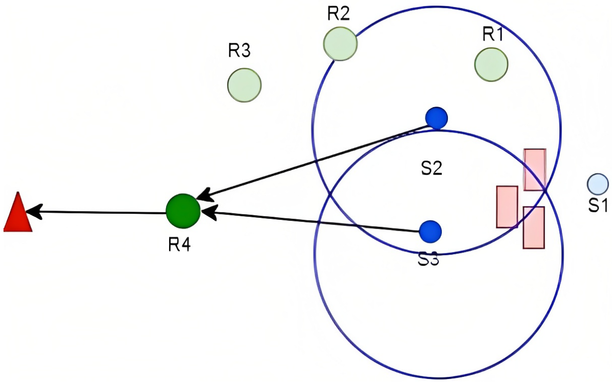
The simultaneous deployment technique, as depicted in Figure 1, offers multiple solutions, including those achieved through sequential deployment. However, due to the NP-Hard nature of the large-scale problem, it becomes time-consuming. To address this challenge, we propose a meta-heuristic-based simultaneous node deployment using the Weighted Sum Method (WSM) [102] formulation. This method ensures a minimal Network Diameter (ND) and network robustness, even if K-1 SNs fail. Notably, our suggested technique optimizes energy consumption



(A) The input CPS and CPR :
3 nodes ($S_i \in$ CPS), i=1...3,
4 nodes ($R_j \in$ CPR), j=1...4,
and one sink node.



(B) Optimal solution 1 : deployed 1 SN,
4 RNs, and 4 hops from the deployed SNs
to the sink node.



(C) Optimal solution 2 : deployed 2 SNs,
1 RN and 1 hop from the deployed SNs
to the sink node.

FIGURE 6.1 – The input CPS and CPR and the obtained optimal solutions.

by excluding SNs from data routing and implementing a two-tiered architecture [91]. Motivated by the effectiveness of meta-heuristics in various domains, including the traveling salesman problem, community detection, email spam detection, and intrusion detection systems, our proposed algorithm provides a balance between computational efficiency and solution quality.

6.2 Problem Modeling

The problem addressed in this research is the deployment of WSNs in Smart Car Parks (SCPs) for surveillance applications, where nodes with limited computational and communication resources monitor the environment. In traditional WSN deployment techniques, SNs and RNs are sequentially placed. Firstly, optimal SN positions are determined to cover target points, followed by optimal RN positions to maintain connectivity between SNs and a sink node. However, this sequential technique has limitations, potentially leading to suboptimal solutions.

The primary issue stems from the fact that optimizing SN placement without considering RN placement may result in suboptimal connectivity to the sink node. Conversely, optimizing RN placement may require additional SNs, increasing deployment costs and Network Diameter (ND). The deployment problem is proven to be NP-Hard, adding complexity to finding an efficient and cost-effective solution.

To overcome these limitations, this research proposes a simultaneous placement technique, considering both SNs and RNs together. The objective is to identify optimal positions for SNs and RNs, taking into account their mutual dependencies. The simultaneous deployment aims to ensure coverage of target points and connectivity of SNs and RNs to the sink node with minimal ND. The research argues that this simultaneous technique is more efficient and cost-effective than the widely used sequential techniques, especially crucial for SCP applications.

The problem statement revolves around optimizing the deployment of WSNs in SCPs by simultaneously considering the placement of SNs and RNs. The goal is to achieve efficient and cost-effective deployment while ensuring coverage of target points and connectivity to the sink node with minimal ND. The research proposes a meta-heuristic-based solution to address the computational complexity of the simultaneous deployment problem.

6.3 Contributions

This work introduces a novel technique to address the simultaneous placement problem in WSNs through the formulation of a Multi-objective Linear Programming (MOLP) model. For small-scale instances, the MOLP model proves effective in finding optimal solutions within reasonable timeframes. However, as the problem scales up, the study presents an innovative solution, the Greedy Chaos Whale Optimization Algorithm (GCWOA). This algorithm combines a Greedy Algorithm for generating initial solutions with Chaos Local Search (CLS) and Whale Optimization Algorithm (WOA) in a hybridized manner. This hybridization enhances the exploration and exploitation capabilities of the optimization process, overcoming the limitations of the traditional WOA, such as vulnerability to local optima.

Key contributions of this work include the development of a fitness function that concurrently addresses multiple objectives, specifically minimizing the number of SNs, RNs, and ND. The formulation of the MOLP model as a mono-objective weighted sum for smaller instances, coupled with the introduction of a novel greedy algorithm and its hybridization with Chaos Local Search, stands out as a unique and comprehensive solution to the deployment optimization problem in WSNs. Through extensive experiments, the study showcases the effectiveness of the proposed GCWOA in solving large instances of the deployment optimization problem, comparing favorably against state-of-the-art meta-heuristics such as WOA, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO). The inclusion of time complexity analysis adds a valuable dimension to the evaluation, demonstrating the practicality and efficiency of the proposed algorithm in real-world scenarios.

6.4 System Model and Problem Formulation

In this section, we present the main components of our network and its operation, along with the sensing and communication models. We then describe the discretization and the MOLP-based formulation of the optimization of the simultaneous SNs and RNs deployment problem. For the sake of readability and clarity, we have listed the symbols and mathematical notations used in this study in Table 6.1.

TABLE 6.1 – Symbols and mathematical notations used in the research study.

Symbols	Description
IoT	Internet of Things
WSN	Wireless Sensor Network
SCP	Smart Car Park
SN	Sensor Node
RN	Relay Node
WOA	Whale Optimization Algorithm
CLS	Chaos Local Search
CWOA	Chaos Whale Optimization Algorithm
GCWOA	Greedy Chaos Whale Optimization Algorithm
MOLP	Multi-objective Linear Programming
BFS	Breadth-First Search
WSM	Weighted Sum Method
RNP	Relay Node Placement Problem
ND	Network Diameter
R_s	Sensing range of SNs
R_{rn} and R_{sn}	Communication range of RNs and SNs, respectively.
T	Set of target points
$Targets\_cover(i)$	Set of target points covered by node $i \in CPS$
CPS and CPR	Candidate positions to place SN and RN, respectively.
V_s and V_r	Set of CPS and CPR, respectively.
V_{cps}^t	Subset of V_s that can cover the target point t.
V_i	Subset of V_r or the sink node which can communicate directly with node i ($i \in (V_s \cup V_r)$)
V_1	Set of $(V_s \cup V_r)$ that can communicate directly with the sink node.
L and W	Number of lines and columns of the grid, respectively.
Q	Size of the mesh grid

6.4.1 System Model

In order to simplify the problem analysis, the following assumptions have been used; it is worth noting that such assumptions have been widely used in similar existing studies [48] [101] [103]

— WSN nodes are equipped with different amounts of energy where the SNs have the smallest amount of energy, then the RNs with a medium amount of energy, and finally the sink node with the largest amount of energy, resulting in heterogeneity.

— RNs have a greater or equal communication range than SNs ($R_{rn} \geq R_{sn}$).

— SNs have the same sensing range R_s.

— Target points and the network nodes are stationary (i.e., do not move).

— Communication takes place over short-range wireless links, where nodes can only communicate if they are within the communication range of one another.

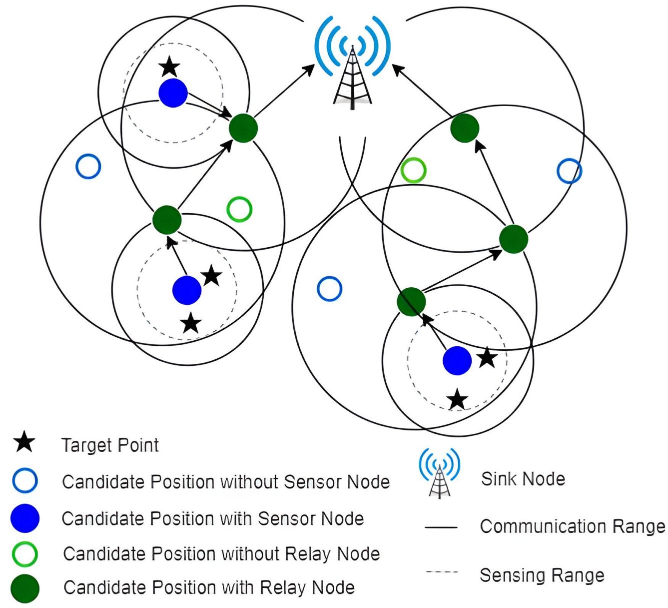


FIGURE 6.2 – Network model instance.

— A perfect communication schedule scheme exists where delays or packet drops due to collisions, queuing, or congestion does not occur.

The WSN comprises heterogeneous nodes, including SNs, RNs, and a singular sink node. The SNs are randomly distributed within the parking area with a set of target points denoted as T. The SNs serve the purpose of sensing predefined locations of the target points and transmitting the collected data either directly to the sink node or via RNs. RNs assist SNs in delivering their data to the sink node, which, in turn, plays the crucial role of disseminating the received data to the external environment.

A specific SN can cover one or more target points only if they fall within its sensing range R_s. Additionally, the current research adopts a two-tiered architecture, where SNs constitute the lower tier, and RNs form the upper tier. The measurement of end-to-end delays between a given SN and the sink node is facilitated by the use of hop count. Similar to the methodology employed in [67], data collection operations are organized into periodic rounds. In each round of data collection, SNs monitor target points and report their findings to the sink node. The network model for the proposed work is illustrated in Figure 6.2.

6.4.2 Sensing Model

In this work, the model can be used for multiple SCP surveillance applications that require the coverage of cars inside the SCP. For example, in cars, fire surveillance applications, i.e., if a fire breaks out in a car, an SN detects that and informs the security office. In car motion surveillance applications, i.e., when a car moves from its spot, an SN detects that and informs the car owner. To simplify the analysis, we adopt the disk sensing model, as described in [104] and [105]. When an event occurs, such as a fire breakout, at a distance d from an SN, the detection probability is as follows :

$$P_{detect}(d) = \begin{cases} 1, & \text{if } d \leq R_s \\ 0, & \text{otherwise.} \end{cases} \tag{6.1}$$

6.4.3 Communication Model

A WSN is considered, where the communication ranges of RNs and SNs are denoted by R_{rn} and R_{sn}, respectively. Typically, R_{rn} is greater than or equal to R_{sn}, and the communication range of the sink node is greater than or equal to R_{rn}. The communication model follows the disk communication model described in [106] and [103]. In this model, communication between an SN and another node is established only when the Euclidean distance d between them is smaller or equal to R_{sn}. Similarly, communication among RNs occurs only when the distance d is smaller or equal to R_{rn}. The communication probability is defined as follows.

$$P_{com_{ij}}(d) = \begin{cases} 1, & \text{if } d \leq R_{sn},\ i \text{ or } j \text{ is SN.} \\ 1, & \text{if } d \leq R_{rn},\ i \text{ and } j \text{ are RN.} \\ 0, & \text{otherwise.} \end{cases} \tag{6.2}$$

6.4.4 Problem Formulation

As shown in Figure 6.3, the region of interest is illustrated as a two-dimensional grid $L \times W$, where L represents the number of lines and W denotes the number of columns. Each mesh grid possesses a uniform length Q following the approach in [104]. Randomly distributed within the grid are target points with known coordinates. Each square cell in the grid has four corners designated as "Candidate Positions for SNs" (CPS) and a center designated as "Candidate Position for RNs" (CPR). The sink node is positioned at a randomly selected location from the CPR set. The primary objective of this research is to simultaneously identify the minimal CPS and CPR, constructing a network with minimal numbers of SNs, RNs, and ND (Nodes in general). This construction must adhere to the constraints of target point coverage, network connectivity, and the two-tiered architecture.

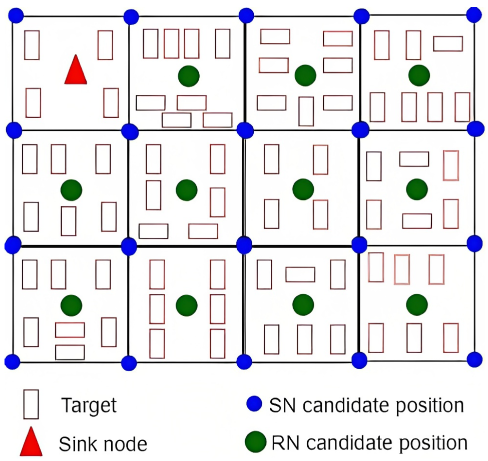


FIGURE 6.3 – Discretization of the deployment space in the 3×4 grid with 20 $CPS \in V_s$, 11 $CPR \in V_r$, 1 sink node and 73 target points.

Based on the linear programming model proposed in [107] that considers only the placement of SNs and taking one objective function, which is the minimization of the number of SNs, in our previous work [77], we have proposed a new version which is a MOLP model which formulates the problem of SNs and RNs deployment simultaneously. Moreover, the objective function minimizes the number of SNs, RNs, and ND while fulfilling the constraints of target points coverage, network connectivity, and two-tiered architecture. In [77], the MOLP model was solved using the $\varepsilon-$constraint method [108]. In the present study, we add the constraint of the K-coverage to provide redundancy in terms of SNs that cover the same target point, then reformulate the MOLP into a mono-objective weighted sum of the three objective functions using WSM.

***Multi-objective Linear Programming* (MOLP)**
The parameters and decision variables used in the MOLP model are listed below in Table 6.2.

TABLE 6.2 – Parameters and decision variables.

Parameter	**Definition**
H	Maximum hop-count allowed
K	Minimal number of SNs required to cover each target point
Decision variables	
x_i^h	Set to 1, if an SN is placed at the position$i \in V_s$, and situated at h hops from the sink node. Is set to 0 otherwise.
r_i^h	Set to 1, if an RN is placed at the position$i \in V_r$, and situated at h hops from the sink node. Is set to 0 otherwise.
y_h	Set to 1, if there is a path of length h hops from the selected RN to the sink node. Is set to 0 otherwise.

The linear programming model that minimizes the objective function F is formulated as follows :

$$F = \left\{ \alpha \times (\sum_{i \in V_s} \sum_{h=1}^{H} x_i^h) + \beta \times (\sum_{i \in V_r} \sum_{h=1}^{H} r_i^h) + \gamma \times (\sum_{h=1}^{H} y_h) \right\} \tag{6.3}$$

with : $0 \leq \alpha, \beta, \gamma \leq 1, \qquad And \alpha + \beta + \gamma = 1$

Subject to :

$$\sum_{h=1}^{H+1} \sum_{i \in V_{V_s}^t} x_i^h \geq K \qquad \forall t \in T \tag{6.4}$$

$$x_i^h \leq \sum_{j \in V_i} r_j^{h-1} \qquad \forall i \in V_s \backslash V_1, \forall h = 2 \dots H \tag{6.5}$$

$$r_i^h \leq \sum_{j \in V_i} r_j^{h-1} \qquad \forall i \in V_r \backslash V_1, \quad \forall h = 2 \dots H \tag{6.6}$$

$$x_i^1 = 0 \qquad \forall i \in V_s \backslash V_1 \tag{6.7}$$

$$r_i^1 = 0 \qquad \forall i \in V_r \backslash V_1 \tag{6.8}$$

$$y_h \geq r_i^h \qquad \forall i \in V_r, \quad \forall h = 1 \dots H \tag{6.9}$$

$$y_h \leq \sum_{i \in V_r} r_i^h \qquad \forall k = 1 \dots H \tag{6.10}$$

$$y_h \leq y_{h-1} \qquad \forall h = 2 \dots H \tag{6.11}$$

The constraint (6.4) ensures that each target point $t \in T$ is covered by at least K SNs. The constraints (6.5) and (6.6) guarantee that each SN and RN deployed at h hops from the sink ($h > 1$) have at least one neighbour RN located at $h-1$ hops from the sink node. Thereby, it satisfies the connectivity between all nodes in the network, thus respecting a two-tiered architecture. The constraints (6.7) and (6.8) avoid the inconsistency of the model, that is to say, no SN and no RN, respectively is neighbour to the sink node unless it is located at 1-hop from it. The constraint (6.9) determines the length of the paths made up of RNs towards the sink node. In other words, if the i^{th} RN located at h hops from the sink, then we mark that there is a path of length h. The constraint (6.10) ensures that if there is a path of length h hops, then there is at least one RN located at h hops from the sink node. This is to avoid the inconsistency of the model. The constraint (6.11) guarantees that the existence of a path of length h hops induces the presence of a path of length $h-1$ hops ($h \geq 2$).

6.5 Whale Optimization Algorithm (WOA)

In the basic WOA meta-heuristic, humpback whales' searching behaviour is mimicked by simulating their movements and sounds. Whales represent solutions to the optimization problem, and their general concept of hunting involves random exploration to locate the prey. When a whale locates its prey, it uses a bubble net attack in which it plunges and swirls around its prey in a spiralling and shrinking circle at the same time. When moving, the whales emit a net of bubbles that trap their prey and disorient it, allowing them to grab it easily. The steps of the WOA are random prey searches and bubble net attacks, as stated in the mathematical models described in the following subsections.

6.5.1 Search for prey

While hunting, humpback whales need to locate their prey. However, it is typically unknown where the prey lies in their search space. For this reason, whales do the exploration based on randomly chosen whale positions. This behaviour can be described by the following equation [109] :

$$D = ||C \cdot X_{rand}(t) - X_i(t)|| \tag{6.12}$$

$$X_i(t+1) = X_{rand}(t) - A \cdot D \tag{6.13}$$

where D is the distance between two whales, $X_i(t)$ is the position vector of the whale i (solution i) at iteration t, $X_{rand}(t)$ is a random whale chosen randomly at iteration t, $||\ ||$ denotes the absolute value and '$\cdot$' denotes an element-by-element multiplication. The rest of the parameters A and C are updated as follow[109] :

$$a = 2 - 2 \cdot \frac{t}{\max\_iteration} \tag{6.14}$$

$$r = Random([0,1]) \tag{6.15}$$

$$A = 2a \cdot r - a \tag{6.16}$$

$$C = 2 \cdot r \tag{6.17}$$

where r is a random number in the range [0, 1], a decreases linearly from 2 to 0 during the iterations, A is a random number in the range $[-a, a]$ and C is a random number in the range [0, 2].

6.5.2 The Bubble Net Attack

A bubble net attack involves two main movements, encircling the prey as depicted in Figure 6.4 and following a spiral-shaped trajectory as illustrated in Figure 6.5.

6.5.3 Encircling Prey

Prey places can be detected by humpback whales, and the whales can encircle them. To imitate this process, the whales move towards the prey (i.e., the current optimal solution) and update its location, assuming the current best solution is the global optimal solution or close to it. As it can be seen in Figure 6.4, exploitation occurs as the value of A decreases ($A < 1$) while exploration occurs as the value of A increases ($A \geq 1$). Therefore, the value of A determines whether exploration or exploitation takes place. In addition, since a decreases linearly from 2 to 0 during iterations and A is a random value in the range $[-a, a]$, once a reaches the value 1, A will be in the range [-1,1]. So the algorithm cannot run the exploration part.

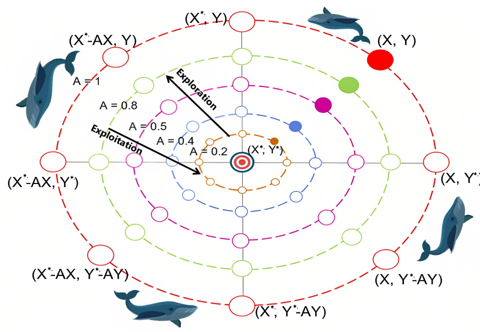


FIGURE 6.4 – Whale encircling prey.

The following equation describes the behaviour of encircling a prey [109] :

$$D = ||C \cdot X^*(t) - X_i(t)|| \tag{6.18}$$

$$X_i(t+1) = X^*(t) - A \cdot D \tag{6.19}$$

where $X^*(t)$ is the best whale (the best solution) until iteration t.

6.5.4 Updating Position along a Spiral-Shaped Trajectory

At this stage, the whale moves around the prey by using a spiral pattern, as shown in Figure 6.5.

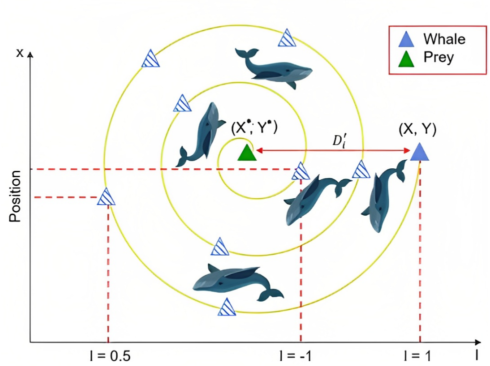


FIGURE 6.5 – Whale moving along a spiral-shaped trajectory.

The following equation describes this behavior[109] :

$$D' = ||X^*(t) - X_i(t)|| \tag{6.20}$$

$$X_i(t+1) = D' \cdot e^{bl} \cdot cos(2\pi l) + X^*(t) \tag{6.21}$$

where D' is the distance between the current best whale and the current whale, b is a constant used to define the logarithmic spiral's shape and l is a random number in [-1,1].

Whales swim around their prey by following a spiral-shaped trajectory as well as a shrinking circle simultaneously. In order to emulate this behaviour, assume that there is a 50% chance that the spiral-shaped trajectory or the shrinking encircling mechanism is used to update the whale position throughout the optimization process. According to [109], the mathematical model for updating the position of whales uses random searches for prey and encircling a prey and spiral-shaped trajectory as follows :

$$X_i(t+1) = \begin{cases} \text{If p} < 0.5 & \begin{cases} \text{If } |\text{A}| \geq 1 & \text{Eq. 6.12 and 6.13} \\ \text{If } |\text{A}| < 1 & \text{Eq. 6.18 and 6.19} \end{cases} \\ \text{If p} \geq 0.5 & \text{Eq. 6.20 and 6.21} \end{cases} \tag{6.22}$$

Where p is a random number in the range [0,1].

To sum up, the pseudo-code of the WOA is given in Algorithm 8.

Algorithm 8 Whale Optimization Algorithm.

Initialize the whales population X_i $(i = 1, 2, ..., n)$
Calculate the fitness of each whale X_i $(i = 1, 2, ..., n)$
Select the best whale (referred to as X^*)
$t \leftarrow 0$
while $t <$ max_iteration **do**
  for each whale X_i **do**
    Determine a, A, C, l, and p
    if $p < 0.5$ **then**
      if $|A| < 1$ (**Exploitation**) **then**
        Update the position of the current whale using Eq. (6.19)
      elseif $|A| \geq 1$ (**Exploration**)
        Select a random whale position (X_{rand})
        Update the position of the current whale using Eq. (6.13)
      end if
    elseif $p \geq 0.5$ (**Exploitation**)
      Update the position of the current whale using Eq. (6.21)
    end if
  end for
  Check if any whale goes beyond the search space and amend it
  Calculate the fitness of the whale
  Update the best whale X^* if there is a better whale in terms of the fitness value
  $t = t + 1$
end while
Return X^*

6.6 Proposed Chaos-WOA with Greedy Solutions Initialization (GCWOA)

This section details the proposed chaos-WOA with greedy solutions initialization and its implementation for the problem of simultaneous deployment of SNs and RNs optimization.

6.6.1 Chaos WOA (CWOA)

Even though WOA has acceptable convergence rates [109], it is still unable to avoid a local optimum, which negatively impacts its convergence rate. In order to overcome this shortcoming and increase efficiency, a chaotic map-based local search referred to as chaos local search (CLS) combined with the basic WOA has been proposed in our previously published conference paper [110].

The utilization of chaos maps in metaheuristic algorithms has demonstrated remarkable benefits in improving their search capabilities and enhancing their overall performance as in [111][112], we have introduced a hybridization between the basic WOA and the CLS based on the Logistic chaos map [113]. The process of hybridization works as follows. After the initialization of all the solutions (whales), the algorithm starts with two sub-populations, which were created by dividing the initial population into two subpopulations. One sub-population adopts the basic WOA, whereas the other sub-population applies the CLS to the best solution obtained so far to exploit its neighbourhood well and escape the local minimum. The Logistic chaos map equation is given by [113] :

$$Logistic(\beta, z(t)) \rightarrow z(t+1) = \beta z(t)(1-z(t)) \tag{6.23}$$

where β is considered to be 4 in Eq. 6.23.

The produced chaotic number $z(t)$ is used in the CLS for updating a random chosen j^{th} dimension of the position of the solution i as follow :

$$X_{ij}(t+1) = lb_j + z(t) \times (ub_j - lb_j) \tag{6.24}$$

where lb_j and ub_j are the lower and upper bounds of the j^{th} dimension, respectively.

Algorithm 9 summarizes the steps of the CWOA. The algorithm starts with the initialization of all the whales (i.e., solutions), and then selects the best whale as the leader of the population. Then, the CWOA divides the population into two subpopulations. For the first sub-population, we assign the position of the leader to each whale and apply CLS by choosing a random dimension j to rotate the current position of the whale along only one selected dimension X_{ij} using equation 6.24. For the second sub-population, we apply the procedure of the basic WOA (Algorithm 8, lines 7-17).

6.6.2 GCWOA-based Deployment in WSNs

In this subsection, we explain the adaptation of the GCWOA to the problem of simultaneous deployment of SNs and RNs optimization in WSNs. We present the solution coding of the problem, the fitness function to evaluate the solutions, the constraints of the problem and the greedy algorithm for population initialization.

Solution Coding

A solution is represented as a vector with a fixed size, which is determined by the size of the CPS + CPR. Each element of the vector is either a $cps \in$ CPS or $cpr \in$ CPR. The value of each

Algorithm 9 Chaos Whale Optimization Algorithm.

Initialize the whales population $X_i(i = 1, 2, ..., n)$
Calculate the fitness of each whale $X_i(i = 1, 2, ..., n)$
Select the best whale (referred to as X^*)
Divide the population into two equal sub-populations S1 and S2
$t \leftarrow 0$
while $t < max\_iteration$ **do**
    for each whale X_i **do**
        if $X_i \in S1$ **then**
            $X_i(t+1) = X^*$
            Choose random j
            Update the current whale position $X_{ij}(t+1)$ using Eq. (6.24)
        elseif $X_i \in S2$
            Apply Algorithm8 (line 7 - 17)
        end if
    end for
    Check if any whale position goes beyond the search space and amend it
    Calculate the fitness of each whale
    Update the best whale X^* if there is a better whale in terms of the fitness value
    $t = t + 1$
end while
Return X^*

element of the vector is set to 1 or 0 to indicate if a *cps* or *cpr* is chosen or not. An example of the solution representation of the example in Figure 6.1 is provided in Table 6.3.

TABLE 6.3 – An example of the solution representation of the problem depicted in Figure 6.1.

	cps_1	cps_2	cpr_1	cps_3	cpr_2	cpr_3	cpr_4
X_1	0	0	0	0	0	0	0
X_2	1	0	1	0	1	1	1
X_3	0	1	0	1	0	0	1

The above line vectors X_i are the solutions encoding of the example in Figure 6.1, where X_1 represents the initial deployment depicted in Figure 6.1a, and X_2 and X_3 are the solution 1 and solution 2 depicted in Figure 6.1b and Figure 6.1c respectively. From these vectors, it can be noted that the values are binary, whereas the WOA is designed to solve problems when the values are continuous. Therefore, in this work, at the end of each iteration of the algorithm, we discretize the vector by taking the value of each element of the vector as a probability. If the value is greater than 0.5 so it will be 1. Similarly, if the value is less than 0.5, so it will be 0.

Fitness Function

The fitness value represents the quality of a solution in relation to the objectives. The proposed work aims to select the minimum number of $cps \in V_s$ and $cpr \in V_r$, respectively, so that all the target points are covered, and all the deployed nodes (SNs and RNs) are connected with the sink node. In what follows, we outline the objectives we use to formulate the fitness function.

1. A minimum number of $cps \in V_s$ should be selected. Hence, the first objective is as follows.

$$F1 = \min \sum_{i \in V_s} X_i \tag{6.25}$$

2. Select a minimum number of $cpr \in V_r$. Therefore, the second objective is

$$F2 = \min \sum_{i \in V_r} X_i \tag{6.26}$$

3. The network should have a minimal ND. Therefore, our third objective is as follows

$$F3 = \min\{\max_{i \in CPS^{sol}} (distance_{(sink,i)}) \tag{6.27}$$

where CPS^{sol} is the set of selected cps of the current solution. The sink is the index of the sink node.

It should be noted that the above objectives conflict with each other. The fitness function in our proposed work is constructed in a way that is conducive to building a trade-off between these conflicting objectives. The multi-objective fitness function is constructed using the WSM, which is a well-known method in multi-objective optimization. Using this method, weight values (α, β, and γ) are multiplied by each objective. As a final step, the multiplied values are added together to create a single objective function, and this can be written as follows.

$$F = \min\{\alpha \times F1 + \beta \times F2 + \gamma \times F3\} \tag{6.28}$$

Where : $0 \leq \alpha, \beta, \gamma \leq 1$, and $\alpha + \beta + \gamma = 1$

Without loss of generality, when using the WSM, it is necessary that all the aggregate objective functions must be in the same range value. Therefore, in this work, we normalize the value of each objective function to be in [0, 1] using the following equation [114].

$$F_i = \frac{F_i - F_i^{LB}}{F_i^{UB} - F_i^{LB}} \tag{6.29}$$

where : F_i^{LB} and F_i^{UB}, are the lower and upper bound of the objective function i, respectively.

Constraints

Most real-world optimization problems must satisfy a number of constraints. In our present work, the constraints which are taken into account are the coverage of the target points, network connectivity, and two-tiered architecture where each SN has a path to the sink node composed only of RNs.

— **Target points coverage :** every target point t must be inside the sensing range of at least K SNs, that can be checked using the following equation :

$$\forall t \in T, \sum_{i \in V^t_{cps}} X_i \geq K \tag{6.30}$$

— **Network connectivity and two-tiered architecture :** each SN must be connected to the sink node directly or through RNs only. This constraint is checked using the Breadth First Search (BFS) algorithm [115] with tabu nodes, where the network is traversed by the algorithm in a breadthward direction starting from the sink node to all the SNs. The following is a mathematical formula for checking this constraint :

$$\forall i \in V_s, \exists Path_{sink,i}, where\{Path_{sink,i} \cap V_s\} = 1 \tag{6.31}$$

Greedy Initial Solutions Generation

In the basic WOA, the initial population is a set of N initial solutions. Each one is a vector containing a finite number of elements. In our case, a solution is a vector of $||V_s + V_r||$ elements. Algorithm 10 describes the process of one solution generation. Firstly, we choose the $cps \in V_s$ that covers the maximum number of target points, place SN in the cps, then remove from the list of target points T the set of target points covered by the deployed SNs and repeat these steps until all target points are covered. Secondly, we randomly choose one deployed SN denoted i, find the shortest path PT from the SN i and the nearest node $j \in \{V_r \cup \{\text{sink}\}\}$, and deploy all the RNs belonging the PT, repeat these steps until all deployed SNs are connected.

The general steps of the GCWOA for WSN deployment are outlined in Figure 6.6.

— **Step 1 :** Initialize α, β, γ, chaos map and all parameters of the WOA.

— **Step 2 :** Generate initial population (set of initial solutions) using algorithm 10.

— **Step 3 :** Check the constraints for all the population solutions.

— **Step 4 :** Evaluate each solution using equation (6.28) and find the best solution (X^*).

— **Step 5 :** Divide the initial population two sub-population.

— **Step 6 :** Update the position of whales belonging to the first sub-population using equation (6.24). Otherwise, update the position of whales using equation (6.22).

— **Step 7** : If the maximum number of iterations is reached, return the current best solution (X^*) as the optimal solution. Otherwise, repeat steps 3 to 7 until the maximum number of iterations is reached.

Algorithm 10 Greedy Initial Solution Generation.

Input : V_s, V_r, T and sink
Output : X vector of $V_s + V_r$
$S = R' = T' = \emptyset$
while $T' \neq T$ **do**
  Choose node $i \in V_s$ that covers the maximum number of target points
  Add i to S
  $T' = T' \cup Targets\_cover(i)$ // $Targets\_cover(i)$ is the set of target points covered by node i
  $V_s = V_s - S$
  for each $j \in V_s$ **do**
    Remove from $Targets\_cover(j)$ the target points that are in T'
  end for
end while
$SN' = S$ // SN' is the set of non connected SNs
while SN' is not empty **do**
  Choose randomly $j \in SN'$
  Remove j from SN'
  Choose the shortest path PT from j to the nearest node $i \in (V_r \cup sink)$
  Add all RNs of the path PT to R'
end while
for $i = 1$ to K **do**
  if $i \in S$ or $i \in R'$ **then**
    $X[i] = 1$
  else
    $X[i] = 0$
  end if
end for
Return X

6.6.3 Computational Time Analysis

Basic steps in the proposed GCWOA-based WSN deployment include the generation of the initial population, the verification of the constraints, the assessment of the fitness calculations, and updating the whale positions.

— The first step in the proposed method is the generation of a population of N whales with size d each. The time complexity of the initial population creation is $\mathcal{O}(N \times C_{initialisation})$, where $C_{initialisation}$ is the time complexity of the greedy solution generation algorithm 10.

— The time complexity to check the coverage constraint for a given whale is $\mathcal{O}(|CPS_{opt}|)$, where $|V_s^{opt}|$ is cardinal of the optimal V_s set and the time complexity to check the connectivity constraint is $\mathcal{O}(C_{BFS})$. (C_{BFS} is the complexity of the BFS algorithm, which is equivalent to $\mathcal{O}(|V_s^{opt}| + |V_r^{opt}| + NB^{edges} + 1)$, where $|V_r^{opt}|$ is cardinal of the optimal V_r set and NB^{edges} is the number of edges).

— The fitness value of each whale in the population can be calculated in two steps, $\mathcal{O}(d)$ to calculate the number of SN and RN deployed and in $\mathcal{O}(C_{BFS})$ to calculate the ND using BFS algorithm.

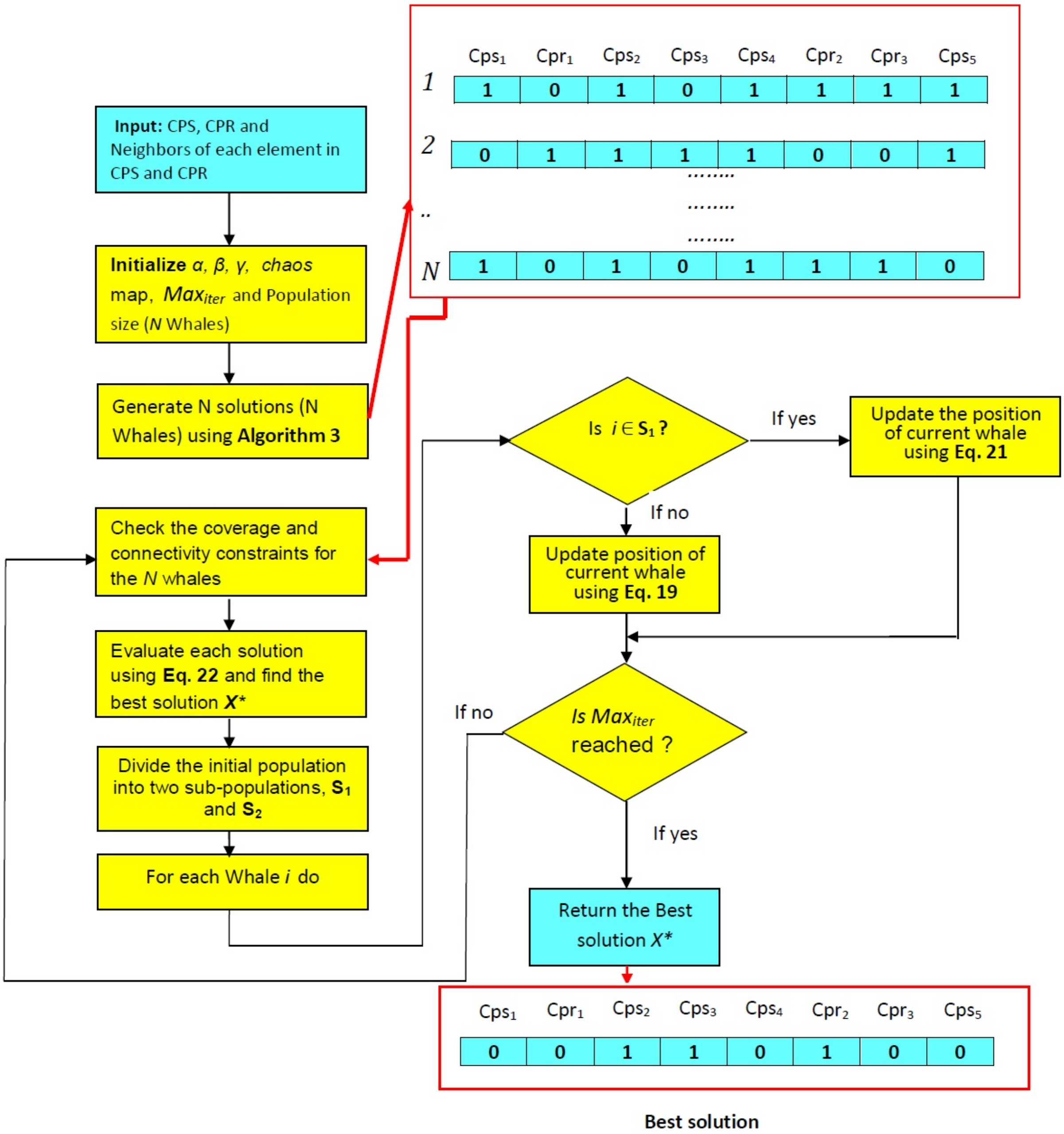


FIGURE 6.6 – Flowchart of the GCWOA for SNs and RNs deployment.

— To update the whale's positions, the algorithm runs for $max_{iteration}$ times **(in Algorithm 9 lines** $6-20$**)**, so the number of iterations is $\mathcal{O}(max_{itereration})$. At each iteration, the algorithm runs N times to update all the population; it takes $\mathcal{O}(max_{itereration} \times N)$. Then to update the dimensions of the whales, the algorithm loops d iterations for each whale, and that only for one sub-population with time complexity of $\mathcal{O}(max_{itereration} \times (N/2) \times d)$, and for the other sub_population the algorithm runs $\frac{N}{2}$ times as it needs to update only one dimension for each whale of the sub-population. Consequently, the overall time complexity of updating the whale position is $\mathcal{O}(max_{iteration} \times \frac{N}{2} \times (d+1))$.

The total time complexity of the proposed GCOWA is the sum of the times taken in each of the individual steps, namely initial population generation, constraints checking, fitness evaluation, and updating the whale positions. This can be expressed as :

$$Total_{complexity} = \mathcal{O}(N \times C_{initialisation} + max_{iteration} \times N \times (|V_s^{opt}| + C_{BFS}) + max_{iteration} \times N \times (d + C_{BFS}) + max_{iteration} \times \frac{N}{2} \times (d+1))$$

$$Total_{complexity} = \mathcal{O}((N \times (C_{initialisation} + max_{iteration} \times (|V_s^{opt}| + 2 \times C_{BFS} + \frac{3\times d+1}{2})))$$

$$Total_{complexity} = \mathcal{O}((N \times (C_{initialisation} + max_{iteration} \times (|V_s^{opt}| + 2 \times (|V_s^{opt}| + |V_r^{opt}| + NB^{edges} + 1) + \frac{3\times d+1}{2})))$$

The time complexity of meta-heuristic algorithms for searching for the global optimal solution can not be determined exactly because these algorithms do not guarantee the finding of the global optimal solution within a given time limit. Nevertheless, in this subsection, we compute the main parameters that affect the total time complexity. It should be noted that the time complexity in our proposal depends mainly on the population size, the maximum number of iterations, the initialization of the solutions population, objective function calculation and check compliance with constraints. Therefore, as the number of candidate positions to place nodes (one decision variable for each candidate position) increases, the time complexity increase.

6.7 Experimental Setup

To demonstrate the effectiveness of our proposed GCWOA, numerous experiments have been conducted on a set of instances with $K \in \{1,2\}$. As in most similar existing studies [104] [75], we have generated instances of the problem with the parameters shown in Table 6.5 and are the same as those in [104]. In order to evaluate the impact of increasing the number of target points, we have randomly scattered a set of target points over the square cells of the grid of size 10x10 in the set of value {100, 200, 300, 400, 500, 600, 700, 800, 900 and 1000}. To evaluate the impact of grid size, we have varied the grid size as in the following set {10x10, 15x15, 20x20 and 25x25} while fixing the number of target points at 1000.

In all the experiments, the sink node position was set to the top right corner of the grid. In order to evaluate performance in terms of the average fitness value (AVG), best fitness value (Best), worst fitness value (Worst), number of SNs (#SN), number of RNs (#RN), ND value (#ND), running times, convergence rate and algorithm stability, we have considered MOLP-based deployment, and the other meta-heuristics based deployments : WOA [116], PSO [117] and GA [48]. In all the experiments, for each instance, each algorithm was run 10 times. For each run, the algorithms generate random starting points (initial solutions) except the GCWOA, which uses our proposed greedy algorithm to generate its starting points. Table 6.5 shows all of the simulation parameters.

Regarding weight factors, various combinations of weight values were tested, with the constraint that the sum of the weights α, β, and γ is equal to 1. Multiple weight values were tested, and the results were tabulated in Table 6.4, where the number of SNs, RNs, and NDs were calculated. The optimal weights were chosen to obtain the solutions of the Pareto front, as represented by the red line in the table. The final set of weights was selected by considering the importance of RNs, which are more costly than SNs, and the significance of ND, particularly in surveillance applications. Based on the table, the optimal combination of weights was found to be $\alpha = 0.1, \beta = 0.5, \gamma = 0.4$. This combination was selected to carry out all further experiments, as it is the best set of weights for the given problem.

TABLE 6.4 – Values for the weights α, β and γ and their corresponding achieved objectives in terms of the number of SNs, RNs, and ND.

weights			1-coverage			2-coverage		
α	β	γ	#SN	#RN	#ND	#SN	#RN	#ND
0,9	0,05	0,05	25	55	13	48	81	13
0,8	0,1	0,1	27	57	13	48	86	13
0,7	0,15	0,15	26	57	13	47	88	14
0,6	0,2	0,2	26	59	12	48	82	13
0,5	0,25	0,25	27	57	13	48	86	13
0,4	0,3	0,3	26	61	13	48	79	13
0,3	0,35	0,35	42	43	13	75	58	13
0,2	0,4	0,4	39	43	13	78	58	13
0,1	0,45	0,45	39	45	13	80	56	13
0,05	0,9	0,05	45	42	16	74	51	16
0,1	0,8	0,1	43	42	15	76	52	15
0,15	0,7	0,15	41	43	13	80	56	13
0,2	0,6	0,2	37	42	13	74	57	13
0,25	0,5	0,25	41	45	13	75	55	13
0,3	0,4	0,3	40	42	13	74	55	13
0,35	0,3	0,35	26	51	13	47	81	13
0,4	0,2	0,4	25	58	12	48	90	13
0,45	0,1	0,45	26	59	13	48	85	13
0,05	0,05	0,9	48	49	12	79	59	13
0,1	0,1	0,8	52	50	12	84	61	13
0,15	0,15	0,7	41	44	13	85	62	13
0,2	0,2	0,6	39	41	13	78	57	13
0,25	0,25	0,5	38	45	13	76	56	13
0,3	0,3	0,4	40	46	13	74	55	13
0,35	0,35	0,3	38	44	13	75	58	13
0,4	0,4	0,2	37	43	13	70	57	13
0,45	0,45	0,1	40	43	14	76	56	13
0,01	0,8	0,19	38	45	13	78	57	13
0,01	0,7	0,29	41	42	13	72	53	13
0,01	0,6	0,39	39	43	13	80	56	13
0,01	0,5	0,46	38	44	13	72	56	13
0,01	0,4	0,59	42	45	13	75	55	13

0,01	0,3	0,69	51	49	12	84	57	13
0,01	0,2	0,79	49	47	12	77	58	13
0,01	0,1	0,89	62	55	12	71	58	13
0,8	0,01	0,19	26	65	13	48	88	13
0,7	0,01	0,29	26	63	13	47	92	13
0,6	0,01	0,39	25	63	13	48	85	13
0,5	0,01	0,46	25	65	13	48	89	13
0,4	0,01	0,59	25	60	13	48	93	13
0,3	0,01	0,69	26	71	12	49	97	13
0,2	0,01	0,79	26	73	12	48	85	13
0,1	0,01	0,89	26	75	12	49	111	12
0,8	0,19	0,01	25	52	15	47	88	13
0,7	0,29	0,01	26	53	16	48	88	13

TABLE 6.5 – Simulation parameters

	Description	Values
System configuration	Processor	i5-8250 U
	RAM	6.00 GB
	Platform	Windows 10 (64-bit)
Simulation tools	Solver	IBM ILOG CPLEX 12.6.1
	Python	python 3.7
Network parameters	L × W	(10x10), (15x15), (20x20), (25x25)
	Q	10 m
	Rs	20 m
	Rsn	20 m
	Rrn	25m
	Number of Targets	100, 200, 300, 400, 500, 600, 700, 800, 900, 1000
GCWOA parameters	Population size	10
	Maximum iterations	250
	a	[2,0]
	b	1
WOA parameters	Population size	10
	Maximum iterations	250
	a	[2,0]
	b	1
GA parameters	Population size	10
	Maximum iterations	250
	Crossover probability	1
	Mutation probability	0.01
PSO parameters	Population size	10
	Maximum iterations	250
	Cognitive component	1
	Social component	0.2
Optimal weights values	α , β and γ	0.1, 0.5 and 0.4

6.8 Experimental Results and Analysis

6.8.1 Impact of the Variation of the Number of Target Points and Grid Size

This subsection examines the impact of the increase in the number of target points and the grid size on deployment costs and ND, where deployment costs are determined by the number of deployed nodes (SNs+RNs).

Table 6.6 shows the deployment costs for both coverage cases, notably 1-Coverage and 2-Coverage while varying the number of target points. It can be seen from the table that the more target points are present in the grid, the more SNs and RNs for coverage and connectivity are required. This can be explained by the limited sensing range of the SNs, which covers a finite set of target points. Therefore, adding more target points implies adding more SNs in order to ensure the coverage of all target points. However, the number of SNs increases more than the number of RNs added to interconnect the new SNs with the sink node. In spite of this, it is clear that when the grid becomes dense with target points, the number of SNs and RNs does not change significantly for both $K = 1$ and $K = 2$. This can be explained by the high number of deployed SNs and RNs that cover the entire grid so that any new target point added to the grid is necessarily covered.

Now, comparing the solutions found by meta-heuristics-based deployment, and MOLP-based deployment, it can be observed from Table 6.6, in any case, in terms of the number of SNs and RNs, the GCWOA provides near-optimal solutions for every problem instance. In terms of the AVG, the Best, and the Worst fitness, the values provided by the GCWOA are close to the optimal solutions of the MOLP and better than those of the other meta-heuristics-based deployment solutions which are almost double for all the problem instances. Furthermore, it should be mentioned that the worst result of the GCWOA is better than the best result of the other competing meta-heuristics and the basic WOA, demonstrating the high performance of the methods added to the proposed GCWOA, namely greedy initial solution generation, CLS and dividing the initial population to benefit from the advantages of the basic WOA.

Table 6.7 shows deployment costs in the case of 1 and 2-coverage based on grid size increments. From Table 6.7, it should be noted that as the grid size increases, the number of SNs, RNs, and ND increases. In fact, when the grid size is small, the 1000 target points are deployed close to each other, so one SN can cover many target points, which implies fewer SNs are needed to ensure the constraint of target points coverage. In contrast, when the grid size becomes large, there is a significant distance between the 1000 target points locations, and with the limited Rs of SNs that require a high number of SNs to cover all the distributed target points mostly when $K = 2$. On the other hand, when the grid size is large, the SNs are located far from the sink node, requiring the placement of a high number of RNs to interconnect all SNs with the sink node, but at the expense of increasing the ND.

Based on the results presented in Table 6.7, it is evident that the MOLP-based deployment is only suitable for the smallest grid, less than 15x15 grid size, as it exhaustively searches through all possible solutions, resulting in significantly longer computation times for larger problem instances. Therefore, it can be argued that the MOLP-based solution is not a scalable option for solving larger grid deployment problems.

For all the considered performance metrics, the GCWOA-based deployment exhibits the best performance for all instances. The GCWOA finds better solutions compared to the PSO, GA, and the basic WOA, with the AVG fitness for $K = 1$ being lower by 23.45%, 28.75%, 26.32% compared to WOA, GA, and PSO, respectively, and for $K = 2$ by 21.33%, 27.42%, 26.82% compared to WOA, GA and PSO respectively. Finally, the obtained results confirm that the

TABLE 6.6 – Grid size $L = 10$ and $W = 10$, AVG fitness, best fitness, worst fitness, number of SNs, number of RNs, and ND value versus number of target points for $K = 1$ and $K = 2$.

Algorithm	K=1						K=2					
	AVG	Best	Worst	#SN	#RN	#ND	AVG	Best	Worst	#SN	#RN	#ND
	100 target points											
WOA	0,25912	0,20193	0,30790	22	35	6	0,35408	0,29256	0,40285	38	51	6
GCWOA	**0,13974**	**0,12973**	**0,16393**	**15**	**13**	**6**	**0,17326**	**0,15058**	**0,18659**	**27**	**17**	**6**
GA	0,31322	0,24654	0,39008	30	45	6	0,36811	0,33129	0,39008	45	53	6
PSO	0,29804	0,25343	0,33306	22	44	6	0,38277	0,34984	0,39787	41	57	6
MOLP		**0,12039**		**10**	**9**	**5**		**0,12948**		**21**	**9**	**5**
	200 target points											
WOA	0,27748	0,23379	0,32810	25	38	6	0,36671	0,30274	0,40193	41	54	5
GCWOA	**0,14569**	**0,12551**	**0,16657**	**16**	**14**	**6**	**0,16986**	**0,15976**	**0,17980**	**31**	**16**	**6**
GA	0,32547	0,27179	0,36244	33	47	6	0,37698	0,33370	0,43875	49	54	6
PSO	0,30906	0,29374	0,34904	25	46	5	0,38462	0,35737	0,40110	44	56	6
MOLP		**0,11699**		**12**	**8**	**5**		**0,13196**		**24**	**9**	**5**
	300 target points											
WOA	0,28384	0,24224	0,35152	27	38	6	0,37601	0,28145	0,42709	45	54	6
GCWOA	**0,15181**	**0,14314**	**0,16657**	**17**	**15**	**6**	**0,17513**	**0,15719**	**0,18825**	**33**	**17**	**6**
GA	0,33408	0,30705	0,37401	37	48	6	0,40189	0,36814	0,43205	48	59	6
PSO	0,31968	0,29456	0,33983	28	47	6	0,39576	0,37668	0,42709	43	59	6
MOLP		**0,12204**		**12**	**9**	**5**		**0,13701**		**24**	**10**	**5**
	400 target points											
WOA	0,28776	0,22287	0,32213	25	40	6	0,39425	0,32782	0,44463	49	58	6
GCWOA	**0,15295**	**0,13469**	**0,16823**	**19**	**15**	**6**	**0,17437**	**0,15976**	**0,18567**	**33**	**17**	**6**
GA	0,33588	0,30028	0,35822	34	49	6	0,40915	0,37071	0,43882	56	60	6
PSO	0,32276	0,28586	0,37089	28	47	6	0,39266	0,35829	0,41368	43	59	6
MOLP		**0,12287**		**13**	**9**	**5**		**0,13361**		**26**	**9**	**5**
	500 target points											
WOA	0,27582	0,24481	0,31552	23	38	6	0,38042	0,33453	0,42452	44	56	6
GCWOA	**0,14669**	**0,13387**	**0,16575**	**18**	**14**	**6**	**0,16967**	**0,15712**	**0,18145**	**31**	**16**	**6**
GA	0,33798	0,31038	0,39091	34	49	6	0,39217	0,36226	0,43774	50	57	6
PSO	0,31901	0,29605	0,34720	31	46	6	0,40849	0,39082	0,43297	46	61	6
MOLP		**0,12287**		**13**	**9**	**5**		**0,13361**		**26**	**9**	**5**
	600 target points											
WOA	0,30507	0,26169	0,35076	29	43	6	0,39189	0,36566	0,42121	46	57	6
GCWOA	**0,15765**	**0,13304**	**0,18239**	**20**	**16**	**6**	**0,18004**	**0,16224**	**0,18733**	**34**	**17**	**6**
GA	0,35125	0,31359	0,40404	40	50	6	0,41531	0,37906	0,44784	56	60	6
PSO	0,33376	0,31625	0,35069	29	49	6	0,42437	0,40009	0,44298	47	63	6
MOLP		**0,12369**		**14**	**9**	**5**		**0,13949**		**27**	**10**	**5**
	700 target points											
WOA	0,31042	0,27568	0,37603	29	44	6	0,38864	0,33949	0,44463	50	56	6
GCWOA	**0,16139**	**0,15390**	**0,16889**	**21**	**15**	**6**	**0,18542**	**0,16646**	**0,19826**	**36**	**18**	**6**
GA	0,35556	0,32287	0,36988	40	51	6	0,40383	0,37732	0,42792	53	58	6
PSO	0,32484	0,28090	0,35161	29	47	6	0,41464	0,38411	0,44812	47	62	6
MOLP		**0,12452**		**15**	**9**	**5**		**0,14031**		**28**	**10**	**5**
	800 target points											
WOA	0,30079	0,26740	0,32312	28	43	6	0,38832	0,36566	0,43230	46	57	6
GCWOA	**0,15298**	**0,13387**	**0,17245**	**19**	**15**	**6**	**0,17419**	**0,15214**	**0,20331**	**35**	**17**	**6**
GA	0,33081	0,29339	0,38354	35	47	6	0,40657	0,37170	0,42624	53	60	6
PSO	0,34291	0,30946	0,38281	31	50	6	0,40975	0,39164	0,42046	47	61	6
MOLP		**0,11864**		**14**	**8**	**5**		**0,13444**		**27**	**9**	**5**
	900 target points											
WOA	0,30944	0,21458	0,38165	29	44	6	0,40154	0,35895	0,46226	49	59	6
GCWOA	**0,15154**	**0,12964**	**0,16657**	**19**	**15**	**6**	**0,18441**	**0,17557**	**0,19495**	**35**	**18**	**6**
GA	0,34593	0,32121	0,38777	38	50	6	0,42128	0,37732	0,44858	56	61	6
PSO	0,32336	0,28777	0,34571	30	48	5	0,41405	0,40604	0,42643	48	62	6
MOLP		**0,12369**		**14**	**9**	**5**		**0,14031**		**28**	**10**	**5**
	1000 target points											
WOA	0,29853	0,22882	0,33306	28	42	6	0,37881	0,33609	0,42525	50	54	6
GCWOA	**0,15802**	**0,14314**	**0,17163**	**19**	**16**	**6**	**0,18474**	**0,16894**	**0,19403**	**36**	**18**	**6**
GA	0,33852	0,30441	0,40266	37	48	6	0,41209	0,38593	0,44040	53	60	6
PSO	0,33393	0,29679	0,37502	32	49	5	0,42109	0,39421	0,44894	48	63	6
MOLP		**0,12452**		**15**	**9**	**5**		**0,14031**		**28**	**10**	**5**

TABLE 6.7 – Number of target points=1000, AVG fitness, best fitness, worst fitness, number of SNs, number of RNs, and ND value versus the grid size for $K = 1$ and $K = 2$.

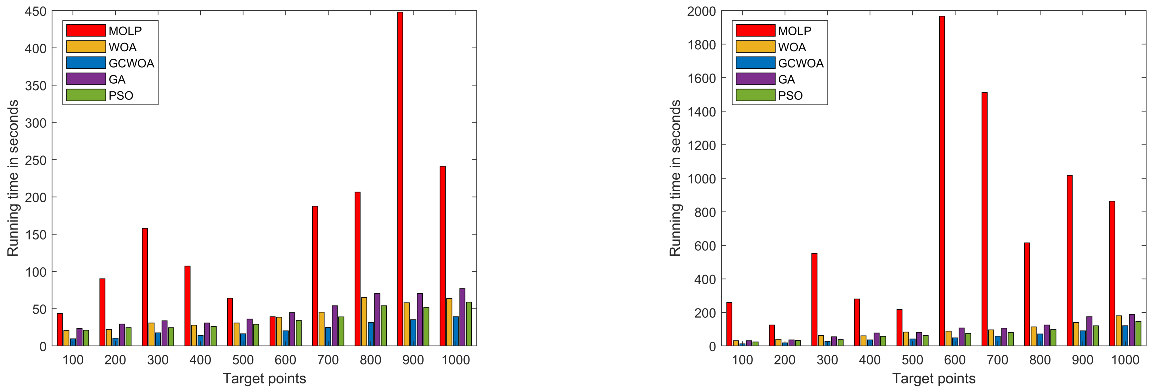
Algorithm	K=1						K=2					
	AVG	Best	Worst	#SN	#RN	#ND	AVG	Best	Worst	#SN	#RN	#ND
	10x10 grid											
WOA	0,29853	0,22882	0,33306	28	42	6	0,37881	0,33609	0,42525	50	54	6
GCWOA	**0,15802**	**0,14314**	**0,17163**	**19**	**16**	**6**	**0,18474**	**0,16894**	**0,19403**	**36**	**18**	**6**
GA	0,33852	0,30441	0,40266	37	48	6	0,41209	0,38593	0,44040	53	60	6
PSO	0,33393	0,29679	0,37502	32	49	5	0,42109	0,39421	0,44894	48	63	6
MOLP		**0,12452**		**15**	**9**	**5**		**0,14031**		**28**	**10**	**5**
	15x15 grid											
WOA	0,42121	0,38741	0,46682	70	127	9	0,49581	0,43289	0,53066	108	155	9
GCWOA	**0,22164**	**0,20750**	**0,22994**	**45**	**44**	**9**	**0,25122**	**0,24126**	**0,26096**	**75**	**52**	**9**
GA	0,46215	0,43635	0,48753	87	145	9	0,52192	0,51225	0,53797	120	165	9
PSO	0,44731	0,42202	0,46777	76	141	9	0,52114	0,49590	0,53630	110	166	9
MOLP	Unresolved due to excessive running time						Unresolved due to excessive running time					
	20x20 grid											
WOA	0,51904	0,48021	0,55014	127	273	12	0,58331	0,56048	0,61249	192	310	12
GCWOA	**0,33770**	**0,32142**	**0,35978**	**107**	**132**	**12**	**0,38473**	**0,36271**	**0,40507**	**166**	**158**	**12**
GA	0,55833	0,52905	0,58789	167	296	12	0,60202	0,58458	0,61659	224	322	12
PSO	0,54157	0,52588	0,55513	138	290	12	0,59736	0,58438	0,61578	200	324	12
MOLP	Unresolved due to excessive running time						Unresolved due to excessive running time					
	25x25 grid											
WOA	0,59520	0,53758	0,63826	209	461	16	0,63258	0,59979	0,65805	293	491	16
GCWOA	**0,47027**	**0,43392**	**0,48486**	**208**	**307**	**16**	**0,51063**	**0,49460**	**0,52250**	**298**	**336**	**16**
GA	0,62821	0,60766	0,64644	261	495	16	0,67293	0,65873	0,68703	355	531	16
PSO	0,61283	0,59362	0,62952	231	488	15	0,66883	0,66084	0,67869	307	535	16
MOLP	Unresolved due to excessive running time						Unresolved due to excessive running time					

proposed GCWOA improves the quality of the solution provided by the basic WOA and can reach better solutions than the well-known meta-heuristics GA and PSO.

For the ND value, from Table 6.6 and Table 6.7, the solutions obtained by the meta-heuristics-based deployment and MOLP-based deployment are close to each other and close to the optimal solution that is found by the MOLP. The reason for this can be explained by the fact that the optimal value for the ND is obtained when all CPS and CPR are chosen to place SNs and RNs, respectively. As a consequence, the best meta-heuristics for the simultaneous deployment of SNs and RNs is the one which finds a minimal number of SNs and RNs without increasing the ND. This is the case shown in Table 6.6 and Table 6.7. The MOLP-based deployment finds a minimal number of SNs and RNs with the lowest ND. While this solution can be applied only for small problem instances, the GCWOA-based deployment is the best for large problem instances as it can find a minimal number of SNs and RNs for the same ND as the other meta-heuristics-based deployment.

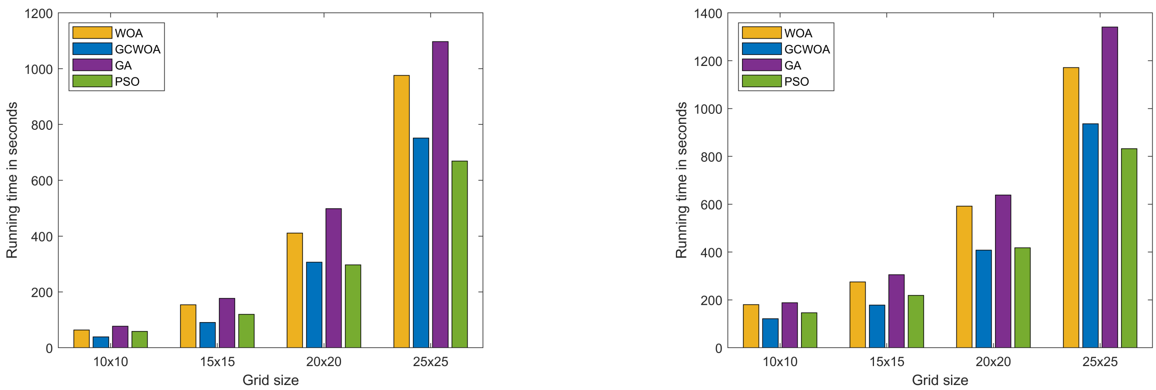
6.8.2 Running Time Analysis

A comparison of the running times of GCWOA, WOA, GA, PSO, and MOLP is shown in Figure 6.7 and Figure 6.8. To obtain these results, we have selected four problem instances, the smallest (10x10 grid and 1000 target points) and the largest (25x25 grid and 1000 target points) with $K = 1$ and $K = 2$. The results in Figure 6.7 and Figure 6.8 reveal that the running time of the meta-heuristics increases as the number of target points, the grid size, or the value of K increases.



(A) (B)

FIGURE 6.7 – Running time versus target points in the 10x10 grid. (a) $K = 1$, (b) $K = 2$.



(A) (B)

FIGURE 6.8 – Running time versus grid size for 1000 target points. (a) $K = 1$, (b) $K = 2$.

From Figures 6.7a and Figure 6.7b, it should be noted that the running time of the MOLP is the highest because the MOLP finds the optimal solutions by exploiting all the solutions search space. On the other hand, the meta-heuristics find near-optimal solutions by exploiting part of the solutions search space, which reduces the running time. As a result, meta-heuristics-based deployment is suitable for large problem instances. Comparing the different meta-heuristics, it is clear that the GCWOA has the lowest running time for both $K = 1$ and $K = 2$, which demonstrates the effectiveness of dividing the population into two sub-populations and the impact of the CLS on the running time, mostly compared to the basic WOA.

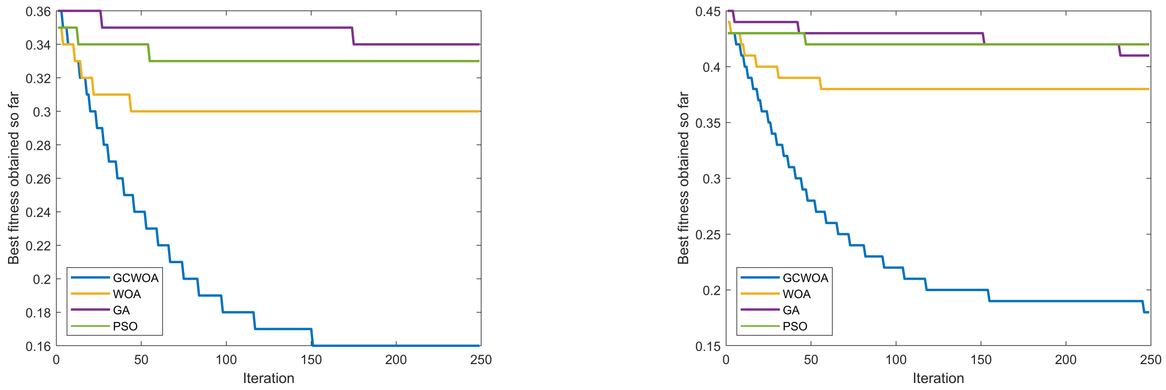
From Figures 6.8a and Figure 6.8b, it can be stated that as the grid size increases, the running time of all the meta-heuristics doubles because the length of the vector of the solution is formed on the basis of the number of CPS and CPR. As a result, as the grid size increases, there are more CPS and CPR, which double the complexity of the problem instance. As far as the running time of the different meta-heuristics is concerned, for the grid sizes 10x10 and 15x15, the GCWOA has the lowest running time. However, for the grid size 20x20, the GCWOA and PSO have comparable running times, while for the grid size 25x25, the PSO becomes the fastest. This implies that as the grid size increases, the PSO is the best in the running time. To conclude, this study has shown that the GCWOA is the most effective meta-heuristic for solving the problem of simultaneous deployment of SNs and RNs in WSNs. The superiority of GCWOA has been confirmed by the results obtained from Tables 6.6 and Table 6.7, as well as Figures 6.7 and 6.8. Additionally, the GCWOA has demonstrated the most optimal solution/running time trade-off for both K=1 and K=2. Overall, these findings suggest that the GCWOA is a highly reliable and efficient method for addressing the simultaneous deployment problem in WSNs, and can be recommended for use in another practical applications.

6.8.3 Convergence Analysis

The convergence results averaged over 10 independent runs have been plotted in Figure 6.9 and Figure 6.10. The selected problem instances are the smallest (10x10 grid using 1000 target points) and the largest (25x25 grid using 1000 target points) with $K \in \{1,2\}$. The figures clearly demonstrate that the proposed GCWOA outperforms the other meta-heuristics and the basic WOA. This is due to the major strength of the GCWOA, which is its ability to avoid local optima and premature convergence, two common weaknesses of the basic WOA method. Furthermore, it can be observed from Figures Figures 6.9a-6.9b that the GCWOA converges early on for the smallest problem instances. Additionally, Figures 6.10a-6.10b demonstrate that the GCWOA displays a stair-step pattern, providing evidence that it is capable of continuously enhancing the current solution. These findings suggest that the proposed GCWOA is able to escape local optima for both small and large problem instances, although it does have a slow convergence rate for large instances that we plan to improve in future work. Overall, the performance results confirm that the GCWOA achieves the primary objective of our research study by providing superior results in solving large problem instances compared to the basic WOA and other state-of-the-art meta-heuristics, such as GA, PSO, and MOLP.

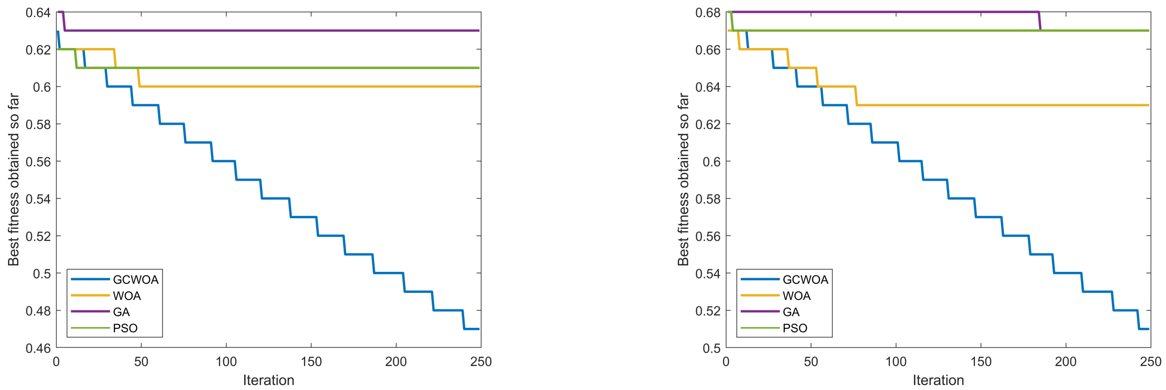
6.8.4 Stability Analysis

We can visualize the distribution of data by means of box plots, typically using three quartiles : the upper, lower, and middle. The points that make up the whisker's edges represent the lowest and highest solutions reached by the meta-heuristics. Lower and upper quartiles are identified by the ends of the rectangles. Generally, narrow box plots indicate strong correlations



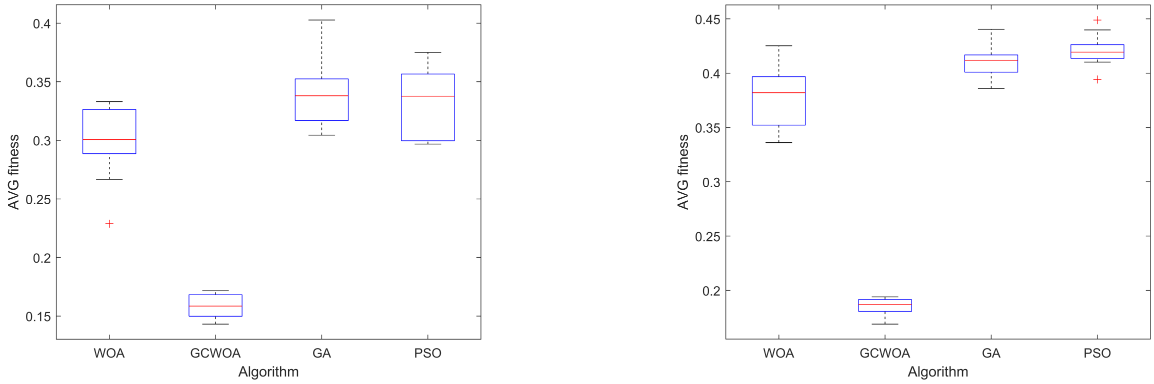
(A) (B)

FIGURE 6.9 – The convergence curve of the average fitness value over 250 iterations for the 10x10 grid and 1000 target points. (a) $K = 1$, (b) $K = 2$.



(A) (B)

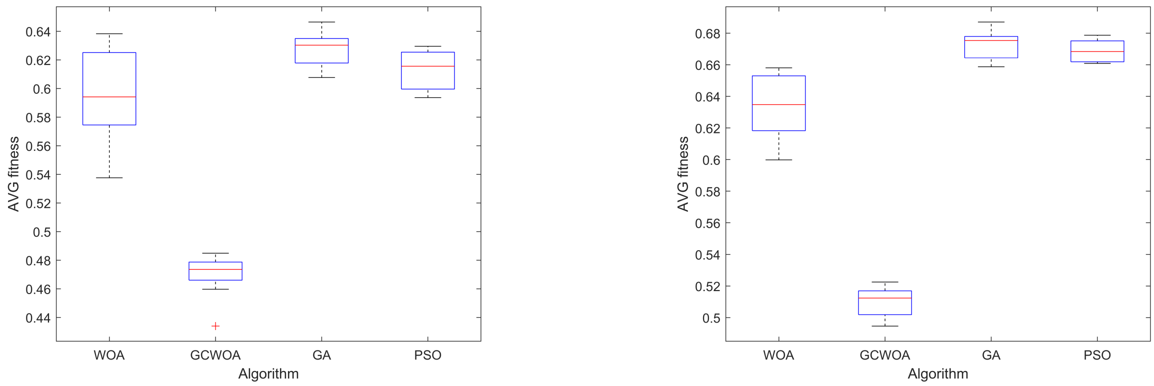
FIGURE 6.10 – The convergence curve of the average fitness value over 250 iterations in the 25x25 grid and 1000 target points. (a) $K = 1$, (b) $K = 2$.



(A) (B)

FIGURE 6.11 – The Box-plot of the average fitness value obtained by each meta-heuristic for a 10x10 grid and 1000 target points. (a) $K = 1$, (b) $K = 2$.

between the solutions. To draw box plots, we have selected four problem instances ; the smallest (10x10 grid using 1000 target points) and the largest (25x25 grid using 1000 target points) with $K \in \{1,2\}$. The results in Figures 6.11a 6.11b 6.12a 6.12b indicate that the proposed GCWOA produces highly narrow box plots in most of the problem instances with the lowest values compared to the other meta-heuristics distributions and the basic WOA, demonstrating the effectiveness of the two added methods, namely greedy algorithm and CLS, to improve the basic WOA. Moreover, it should be noted that PSO and GA always find the worst fitness values, and they possess small box plots for both problem instances when $K = 2$ in Figure 6.11b and Figure 6.12b. This is because they get trapped in the local optimum in the early iteration, and as a result, they cannot manage to improve the initial solution. It is worth mentioning that when the best initial solutions have a near fitness value for the 10 runs, the final solutions are comparable to each other.



(A) (B)

FIGURE 6.12 – The Box-plot of the average fitness value obtained by each meta-heuristic for a 25x25 grid and 1000 target points. (a) $K = 1$, (b) $K = 2$.

6.9 Conclusions

In the investigation of the deployment problem for SCP surveillance using WSNs, a notoriously NP-Hard challenge is addressed. In this chapter, an innovative approach is introduced, characterized by the simultaneous determination of optimal positions for both SNs and RNs, allowing the optimization of multiple objectives, including the numbers of SNs and RNs, along with network diameter (ND). The distinctiveness of this novel method lies in its simultaneous consideration of SN and RN placements, with a focus on coverage and connectivity constraints, setting it apart from conventional sequential techniques. Utilizing a Multi-Objective Linear Programming (MOLP) model for small-scale instances and a sophisticated Greedy Chaos Whale Optimization Algorithm (GCWOA) for larger problem spaces, the NP-Hard nature of the problem is effectively addressed, providing efficient solutions.

The empirical evaluation, encompassing variations in grid sizes and target numbers, revealed the efficacy of GCWOA, particularly in larger grid sizes, where scalability challenges hamper MOLP. Siginificant reductions in the total number of selected nodes compared to traditional algorithms were demonstrated by GCWOA, accompanied by faster convergence rates and competitive running times. Future endeavors will include the enhancement of solution space representation through the utilization of multi-objective optimization, such as MO-WOA, to generate a Pareto front. Additionally, efforts will be directed towards the improvement of the convergence of GCWOA, along with the extension of its application to 3D space scenarios involving mobile targets, reflecting a commitment to advancing the state-of-the-art in WSN deployment optimization for SCP surveillance.

Chapitre 7

Conclusions and Future Research Directions

The optimal deployment of Wireless Sensor Networks (WSNs) for a smart car park (SCP) applications has been addressed in this research work by simultaneously identifying the locations of Sensor Nodes (SNs) and Relay Nodes (RNs). The objective is to minimize the number of SNs, the number of RNs, and the network diameter while satisfying coverage and connectivity constraints. The exploration of WSNs and their deployment optimization by this study has helped to provide a comprehensive understanding of the theoretical foundations, practical considerations, and innovative approaches in the field.

Chapter 1 introduced the essential components of WSNs, including SNs and RNs and their internal architectures. Diverse applications, from environmental monitoring to SCP Surveillance, were highlighted, showcasing the versatility and real-world implications of WSN technology. Communication models, routing, and the different types of WSNs deployment were discussed, providing a bridge between theoretical concepts and practical implementations. The present research work has proposed a novel deployment technique, departing from conventional sequential placement techniques, to simultaneously determine optimal positions for SNs and RNs while minimizing multiple objectives including the network diameter.

Chapter 2 discussed Multi-Objective Optimization and Pareto Multi-Objective Optimization (PMO) problems, categorizing resolution methods into different families. The significance of exact and approximate techniques in operations research was emphasized, with a focus on striking a balance between exploration and exploitation. Heuristic and metaheuristic approaches, such as greedy algorithms and genetic algorithms, enrich the toolkit for tackling PMO, offering a nuanced understanding of resolution methods and their practical applicability.

Chapter 3 conducted a review of the state-of-the-art in the deployment optimization of WSNs for surveillance applications. Multi-objective optimization emerged as a key focus, enabling the simultaneous optimization of conflicting objectives such as minimizing nodes, maximizing coverage, and ensuring network connectivity. Integration with technologies like cameras, drones, and machine learning enriched surveillance capabilities, reflecting the continuous evolution of WSNs applications. One of the most important applications is fire detection.

Chapter 4 addressed the NP-Hard problem of optimizing the deployment of WSNs, specifically focusing on SCPs for surveillance applications. Unlike conventional techniques, the proposed approach simultaneously determines optimal positions of SNs and RNs in a given area while minimizing multiple objectives. Such objectives encompass the number of selected positions to place SNs and RNs, as well as the network diameter, ensuring coverage of each target point by at least one SN and connectivity between the selected position to place SNs and the sink node.

Moreover, Chapter 4 proposed a Multi-Objective Binary integer Linear Programming (MOBILP) model for optimal solutions in smaller problem instances, specifically within a 8x8 grid of 20 square meters each, emphasizing the simultaneous placement of SNs and RNs, wherein the number of each type of node constituted an objective function. The MOBILP-based techniques encompassed mono-objective optimization entailing the simultaneous placement of SNs and RNs, denoted as Technique 1. Subsequently, a second technique, denoted as Technique 2, was implemented, wherein SNs and RNs were deployed sequentially. Specifically, SNs were initially placed to cover designated target points, followed by the subsequent placement of RNs to establish connectivity between the deployed SNs and the sink node.

The primary focus was on evaluating the impact of target density and distribution on the deployment solutions, as well as assessing the trade-offs between conflicting objectives such as the number of RNs, SNs, and the network diameter. The outcomes revealed several key findings. The MOBILP provided a set of alternative solutions, including the solution obtained from Technique 1. Additionally, all solutions from Technique 2 were either included in the MOBILP solution set or dominated by those solutions. This suggested that modeling the simultaneous deployment problem of SNs and RNs as a multi-objective optimization allowed decision-makers more flexibility in choosing optimal trade-offs based on the problem's context. This flexibility was particularly useful in selecting specific numbers of RNs to achieve the minimum associated numbers of SNs and network diameter.

The study also explored the impact of target density on deployment. The results indicated that as the density of targets increased within the smart parking area, more SNs and RNs were required, leading to a significant increase in deployment costs. This was attributed to the limited sensing range of SNs, which reduced coverage and necessitated more nodes to meet coverage and connectivity requirements. Furthermore, the distribution of target positions in different scenarios affected WSNs deployment. For instance, in a scenario where target points were concentrated in one area, the number of SNs was the lowest since one SN could cover multiple target points. The position of the sink node was found to impact only the network diameter, with the optimal position being in the middle of the study area. However, the study noted that MOBILP incurred longer running times compared to the other two techniques. The increased complexity resulted from the simultaneous deployment of SNs and RNs and the multi-objective nature of the problem. For example, in a scenario with 50 target points, the resolution times were 2.67s, 3.02s, and 35.88s for techniques 1, 2, and MOBILP, respectively. As the number of target points increased, MOBILP's running time also increased substantially. Despite the longer running times, the study argued that the resolution times were reasonable as the deployment was considered offline.

For larger instances, up to a 25x25 grid, Chapter 5 introduced an enhanced version of the well-known Whale Optimization Algorithm (WOA), termed the Greedy Chaos Whale Optimization Algorithm (GCWOA). GCWOA integrated a greedy algorithm for the iterative placement of SNs and RNs to generate feasible initial solutions. Subsequently, a Chaos Local Search (CLS) was implemented on one sub-population to exploit their neighborhood, while the WOA was applied to the other sub-population to leverage its strengths in exploration and exploitation.

The impact of varying grid size and the number of target points on the proposed approaches was systematically investigated. The study found that MOBILP was effective for small instances (e.g., a 10x10 grid) even with a substantial number of target points (up to 1000). However, its efficiency diminished for grid sizes equal to or greater than 15x15 due to the exponential increase in decision variables (candidate node positions), resulting in prolonged search times for optimal solutions. Consequently, the proposed GCWOA is positioned as a more compelling solution for deployment problems in larger grid sizes exceeding 15x15.

Simulation results and statistical analysis demonstrated the superiority of GCWOA over conventional algorithms such as WOA, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO). In a 25x25 grid, GCWOA reduced the total number of selected positions to place SNs and RNs by 26.16%, 37.92%, and 33.06%, respectively, compared to WOA, GA, and PSO. This reduction was achieved with a lower running time than WOA and GA, while remaining comparable to PSO. Furthermore, GCWOA exhibited the fastest convergence rate among the algorithms, allowing it to reach superior solutions in early iterations. Overall, the findings underscore the effectiveness of the proposed GCWOA in optimizing the WSN deployment for SCP surveillance applications in larger grid sizes.

Collectively, the thesis work contribute to advancing the understanding and application of WSNs. The theoretical insights, coupled with practical solutions and innovative algorithms, have expanded the scope of WSNs technology. The GCWOA stands out as a promising metaheuristic, showcasing its potential to address the complexities of WSNs deployment. As we conclude this exploration, there is a scope for future advancements, embracing new challenges, and continuing to push the boundaries of technological innovation in the realm of WSNs. The detailed future scope is discussed below.

7.1 Future Research

One immediate future objective is the extension of our study to incorporate multi-objective optimization approaches, such as MO-WOA. This approach aims to provide a Pareto front with diverse and high-quality solutions, offering decision-makers a range of trade-offs among conflicting objectives. The other future objectives are discussed in the following subsections.

Objective Function Refinement : In our current work, the formulation of objective functions relies on the Weighted Sum Method (WSM). Future endeavors will explore alternative optimization approaches, such as multi-objective optimization methods, to refine and enhance the formulation of objective functions. This shift aims to mitigate the need for extensive preliminary simulation runs and improve the overall efficiency of the optimization process.

Exploration of 3D Deployment in SCP with Mobile Targets : The three-dimensional (3D) space introduces additional complexities to the deployment problem, especially in the context of SCP surveillance with mobile targets. Our future research will delve into this dimension, exploring the application of recent evolutionary optimization techniques to address the challenges posed by 3D deployment scenarios.

Enhancement of GCWOA Convergence : Building on the success of the GCWOA, future efforts will focus on enhancing its convergence properties. Strategies to improve the algorithm's ability to reach optimal or near-optimal solutions in early iterations will be explored, contributing to its efficiency and applicability in large-scale deployment scenarios.

Exploration of Emerging Evolutionary Optimization Techniques : Continuing with our exploration of optimization techniques, we are keen on investigating emerging evolutionary optimization methods. These approaches may offer novel methods for solving complex deployment problems and contribute to the advancement of state-of-the-art solutions.

Our current research lays a foundation for addressing the challenges of WSNs deployment in SCPs applications. The identified conclusions and future work directions aim to propel the field forward, fostering innovation and practical applicability in real-world surveillance scenarios. Through continuous exploration and refinement, we anticipate contributing valuable insights and solutions to the evolving landscape of WSNs deployment optimization.

Bibliographie

[1] Hao Wei-Dong and Zhu Bo-Xuan. Smart home wireless system using zigbee and ieee802.15.4. In *2016 Sixth International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC)*, pages 858–863, 2016.

[2] Gang Li, Bin He, Yanmin Zhou, Zhongpan Zhu, and Hongwei Huang. Information granularity with the self-emergence mechanism for event detection in wsn-based tunnel health monitoring. *IEEE Sensors Journal*, 20(15) :8265–8275, 2020.

[3] Samayveer Singh, Aridaman S Nandan, Aruna Malik, Neeraj Kumar, and Ahmed Barnawi. An energy-efficient modified metaheuristic inspired algorithm for disaster management system using wsns. *IEEE Sensors Journal*, 21(13) :15398–15408, 2021.

[4] Zhendong Wang, Huamao Xie, Daojing He, and Sammy Chan. Wireless sensor network deployment optimization based on two flower pollination algorithms. *IEEE Access*, 7 :180590–180608, 2019.

[5] Trista Lin, Hervé Rivano, and Frédéric Le Mouël. A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12) :3229–3253, 2017.

[6] Marwa Fattoum, Zakia Jellali, and Leila Najjar Atallah. A joint clustering and routing algorithm based on ga for multi objective optimization in wsn. In *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)*, pages 1–5, 2020.

[7] Sujee R. and Kannammal K.E. Behavior of leach protocol in heterogeneous and homogeneous environment. In *2015 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–8, 2015.

[8] Akhilesh A Waoo and Sanjay Sharma. Analysis of energy efficient coverage and prolonging lifetime by comparing homogenous and heterogeneous wireless sensor networks. In *2018 International Conference on Advanced Computation and Telecommunication (ICACAT)*, pages 1–4, 2018.

[9] Nitin Kumar, Vinod Kumar, and Pawan Kumar Verma. An energy-efficient wireless sensor networks approach based on adv-leach2 clustering protocols for heterogeneous and homogeneous. In *2021 5th International Conference on Information Systems and Computer Networks (ISCON)*, pages 1–6, 2021.

[10] Swati and Mamta Arora. Several categories of energy-efficient routing protocols, features, and security necessities in wsn : A review. In *2022 3rd International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, pages 1–6, 2022.

[11] Shi Lan, Miao Qilong, and Jinglin Du. Architecture of wireless sensor networks for environmental monitoring. In *2008 International Workshop on Education Technology and Training 2008 International Workshop on Geoscience and Remote Sensing*, volume 1, pages 579–582, 2008.

[12] Pooja Krishnath Patil and S.R. Patil. Review on structural health monitoring system using wsn for bridges. In *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, volume 1, pages 628–631, 2017.

[13] Mosad Alkhathami, Lubna Alazzawi, and Ali Elkateeb. Large scale border security systems modeling and simulation with opnet. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1–8, 2017.

[14] T. Lin, H. Rivano, and F. Le Mouël. A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12) :3229–3253, 2017.

[15] Yazeed AlZahrani, Jun Shen, and Jun Yan. Spatial goal refinement patterns for iot applications. In *2022 IEEE International Conference on Edge Computing and Communications (EDGE)*, pages 50–59, 2022.

[16] Riham Elhabyan, Wei Shi, and Marc St-Hilaire. Coverage protocols for wireless sensor networks : Review and future directions. *Journal of Communications and Networks*, 21(1) :45–60, 2019.

[17] Riham Elhabyan, Wei Shi, and Marc St-Hilaire. Coverage protocols for wireless sensor networks : Review and future directions. *Journal of Communications and Networks*, 21(1) :45–60, 2019.

[18] Sandeep Kaur and Ruby Goel. A review on data transmission techniques for energy efficiency in wireless sensor networks. In *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 699–703, 2016.

[19] K. Norton. Point-to-point radio relaying via the scatter mode of tropospheric propogation. *IRE Transactions on Communications Systems*, 4(1) :39–49, 1956.

[20] Daniel Czech, Amit Kumar Mishra, and Michael Inggs. Identifying radio frequency interference with hidden markov models. In *2016 Radio Frequency Interference (RFI)*, pages 21–25, 2016.

[21] Yogesh Patidar, Manish Jain, and Ajay Kumar Vyas. Modified unexpended energy based stable cluster head selection for homogeneous wireless sensor networks (wsns). In *2022 3rd International Conference on Computing, Analytics and Networks (ICAN)*, pages 1–5, 2022.

[22] Zheng Gengzhong and Liu Qiumei. A survey on the topology of wireless sensor networks based on small world network model. In *2010 2nd International Conference on Future Computer and Communication*, volume 1, pages V1–67–V1–71, 2010.

[23] Manoj Tolani, Ambar Bajpai, Sunny, Rajat Kumar Singh, Lunchakorn Wuttisittikulkij, and Piya Kovintavewat. Energy efficient hybrid medium access control protocol for wireless sensor network. In *2021 36th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, pages 1–4, 2021.

[24] Kristóf Fodor and Attila Vidács. Efficient routing to mobile sinks in wireless sensor networks. 10 2007.

[25] Joseph Haule and Kisangiri Michael. Deployment of wireless sensor networks (wsn) in automated irrigation management and scheduling systems : a review. In *Proceedings of the 2nd Pan African International Conference on Science, Computing and Telecommunications (PACT 2014)*, pages 86–91, 2014.

[26] Zhixiao Wang and Abdul Hadi. Revisiting the state-of-the-art deployment and coverage preserving performance for wsn. In *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 557–562, 2019.

[27] Mohammed Y. M Alnaham and Iman Abuel Maaly Abdelrahaman. Improving qos in wsn based on an optimal path from source to sink node routing algorithm. In *2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, pages 1–6, 2018.

[28] Bishad Ghimire, Ausif Mahmood, and Khaled Elleithy. Hybrid quantum approximate optimization using enhanced ant colony optimization to solve large-scale combinatorial optimization problems. In *2021 8th International Conference on Soft Computing Machine Intelligence (ISCMI)*, pages 108–113, 2021.

[29] Keiichiro Yasuda, Junichi Tsuchiya, Kenichi Tamura, and Yuta Obinata. Combinatorial optimization method using distance in scheduling problem. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1195–1200, 2018.

[30] Wen Chuanjun, Wang Ling, and Ren Xuejing. General particle swarm optimization algorithm. In *2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, pages 1204–1208, 2023.

[31] Oct 2023.

[32] Shuning Yang, Zhongping Yang, Jianyun Zhang, and Qingsong Zhou. A subarray level adaptive interference suppression method based on particle swarm optimization and greedy algorithm. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1124–1129, 2020.

[33] Yashashree A. Jakhade, Nitin R. Gavai, and G. V. Garje. Tree based aggregation in wsn for preventing compromised nodes. In *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, pages 1–5, 2017.

[34] Hanan Alsouly, Michael Kirley, and Mario Andrés Muñoz. An instance space analysis of constrained multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 27(5) :1427–1439, 2023.

[35] Meng Xia, Qiang Zhou, Jan Sykulski, Shiyou Yang, and Yanhong Ma. A multi-objective topology optimization methodology based on pareto optimal min-cut. *IEEE Transactions on Magnetics*, 56(3) :1–5, 2020.

[36] https://engineering.purdue.edu/~sudhoff/ee630/Lecture09.pdf. [Accessed 17-12-2023].

[37] A multi-objective optimization approach for the group formation problem. In *Expert Systems with Applications*, volume 162, page 113828, 2020.

[38] Nyoman Gunantara. A review of multi-objective optimization : Methods and its applications. *Cogent Engineering*, 5(1) :1502242, 2018.

[39] Pablo A. Miranda-Gonzalez, Javier Maturana-Ross, Carola A. Blazquez, and Guillermo Cabrera-Guerrero. Exact formulation and analysis for the bi-objective insular traveling salesman problem. *Mathematics*, 9(21), 2021.

[40] Tomas Valencia Zuluaga, Amelia Musselman, and Ignacio Aravena. Deterministic dual dynamic programming applied to optimal power system restoration. In *2022 IEEE Power Energy Society General Meeting (PESGM)*, pages 1–5, 2022.

[41] Yutika Amelia Effendi, Fitri Retrialisca, and Nania Nuzulita. Conformance checking to evaluate business process models using modified time-based heuristics miner algorithm. In *2019 IEEE Conference on Sustainable Utilization and Development in Engineering and Technologies (CSUDET)*, pages 81–86, 2019.

[42] WK Wong and Chew Ing Ming. A review on metaheuristic algorithms : Recent trends, benchmarking and applications. In *2019 7th International Conference on Smart Computing Communications (ICSCC)*, pages 1–5, 2019.

[43] P. Slavík. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms*, 25(2) :237–254, 1997.

[44] Ning Wang, Lili Yu, Linxiao Yu, and Shanhui Ke. Optimization of containership stowage in circular route based on greedy algorithm. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 146–150, 2020.

[45] W. Zeng and R. L. Church. Finding shortest paths on real road networks : The case for A. *International Journal of Geographical Information Science*, 23(4) :531–543, 2009.

[46] Herri Akhmad Bukhori, Tiksno Widyatmoko, S Sunarti, Irawan Dwi Wahyono, Amira Eza Febrian Putri, Dwi Sudarno Putra, Ahmad Awaluddin Baiti, Muhammad Zenda Rud, and Diana Putri. Optimizing tabu list in tabu search algorithm for efficient and optimal scheduling. In *2023 Sixth International Conference on Vocational Education and Electrical Engineering (ICVEE)*, pages 73–76, 2023.

[47] Sirui Zheng, Xiangjing Lai, and Wei Gong. Neighborhood decomposition based variable neighborhood search for the capacitated p-median problem. In *2021 China Automation Congress (CAC)*, pages 1101–1105, 2021.

[48] Suneet Kumar Gupta, Pratyay Kuila, and Prasanta K. Jana. Genetic algorithm approach for k-coverage and m-connected node placement in target based wireless sensor networks. *Computers & Electrical Engineering*, pages 544–556, 2016.

[49] Ivan Cruz-Aceves, Fernando Cervantes-Sanchez, Arturo Hernandez-Aguirre, Martha A. Hernández-González, and Sergio Solorio-Meza. Chapter 6 - binary descriptor design for the automatic detection of coronary arteries using metaheuristics. In Tapan Gandhi, Siddhartha Bhattacharyya, Sourav De, Debanjan Konar, and Sandip Dey, editors, *Advanced Machine Vision Paradigms for Medical Image Analysis*, Hybrid Computational Intelligence for Pattern Analysis and Understanding, pages 169–188. Academic Press, 2021.

[50] Alper Kerem, Ali Saygin, and Rasoul Rahmani. Wind power forecasting using a new and robust hybrid metaheuristic approach : A case study of multiple locations. In *2019 19th International Symposium on Electromagnetic Fields in Mechatronics, Electrical and Electronic Engineering (ISEF)*, pages 1–2, 2019.

[51] Saunhita Sapre and S Mini. Optimized relay nodes positioning to achieve full connectivity in wireless sensor networks. *Wireless Personal Communications*, 99(4) :1521–1540, 2018.

[52] I. Chukwuemeka Chimsom and Maki K. Habib. Design of a two-tier wsn-based iot surveillance system with cloud integration. In *2019 20th International Conference on Research and Education in Mechatronics (REM)*, pages 1–7, 2019.

[53] Slaheddine Chelbi, Habib Dhahri, and Rafik Bouaziz. Node placement optimization using particle swarm optimization and iterated local search algorithm in wireless sensor networks. *International Journal of Communication Systems*, 34(9) :e4813, 2021.

[54] Ahmed Musa, Virigilio Gonzalez, and Dante Barragan. A new strategy to optimize the sensors placement in wireless sensor networks. *Journal of Ambient Intelligence and Humanized Computing*, 10 :1389–1399, 2019.

[55] Yu Han, Gang Li, Rui Xu, Jian Su, Jian Li, and Guangjun Wen. Clustering the wireless sensor networks : A meta-heuristic approach. *IEEE Access*, 8 :214551–214564, 2020.

[56] Palvinder Singh Mann and Satvir Singh. Improved metaheuristic based energy-efficient clustering protocol for wireless sensor networks. *Engineering Applications of Artificial Intelligence*, 57 :142–152, 2017.

[57] Farzad Kiani, Amir Seyyedabbasi, and Sajjad Nematzadeh. Improving the performance of hierarchical wireless sensor networks using the metaheuristic algorithms : efficient cluster head selection. *Sensor Review*, 41(4) :368–381, 2021.

[58] Alanoud Alotaibi and Farrukh Nadeem. A review of applications of linear programming to optimize agricultural solutions. *International Journal of Information Engineering and Electronic Business*, 15(2) :11, 2021.

[59] S Kokilavani Sankaralingam, AS Narmadha, et al. Energy aware decision stump linear programming boosting node classification based data aggregation in wsn. *Computer Communications*, 155 :133–142, 2020.

[60] Bo Yuan, Huanhuan Chen, and Xin Yao. Optimal relay placement for lifetime maximization in wireless underground sensor networks. *Information Sciences*, 418 :463–479, 2017.

[61] Qianqian Yu, Chen Yang, Guangming Dai, Lei Peng, and Xiaoyu Chen. Synchronous wireless sensor and sink placement method using dual-population co-evolutionary constrained multiobjective optimization algorithm. *IEEE Transactions on Industrial Informatics*, 19(6) :7561–7571, 2022.

[62] Jilong Sun and Ronghe Wang. Sensors layout optimization based on cluster analysis in water supply network. In *Proc. 3rd Int. Conf. Eng. Technol. Appl.(ICETA)*, pages 1011–1015, 2016.

[63] Errol L Lloyd and Guoliang Xue. Relay node placement in wireless sensor networks. *IEEE Transactions on computers*, 56(1) :134–138, 2006.

[64] Suman Arora, Geeta Nijhawan, Gourav Verma, and Ruchi J Patel. A systematic survey on various energy harvesting systems for wsn applications. In *2021 International Conference on Industrial Electronics Research and Applications (ICIERA)*, pages 1–5, 2021.

[65] Pothuri Surendra Varma and Veena Anand. Indoor localization for iot applications : Review, challenges and manual site survey approach. In *2021 IEEE Bombay Section Signature Conference (IBSSC)*, pages 1–6, 2021.

[66] Soumaya Fellah and Mejdi Kaddour. Exact and efficient heuristic deployment in wsn under coverage, connectivity, and lifetime constraints. *International Journal of Mobile Computing and Multimedia Communications (IJMCMC)*, 8(2), 2017.

[67] Govind P Gupta and Sonu Jha. Biogeography-based optimization scheme for solving the coverage and connected node placement problem for wireless sensor networks. *Wireless Networks*, 25(6) :3167–3177, 2019.

[68] Chun-Wei Tsai, Pei-Wei Tsai, Jeng-Shyang Pan, and Han-Chieh Chao. Metaheuristics for the deployment problem of wsn : A review. *Microprocessors and Microsystems*, 39(8) :1305–1317, 2015.

[69] Sinem Akyol and Bilal Alatas. Plant intelligence based metaheuristic optimization algorithms. *Artificial Intelligence Review*, 47 :417–462, 2017.

[70] Kavita Jaiswal and Veena Anand. A qos aware optimal node deployment in wireless sensor network using grey wolf optimization approach for iot applications. *Telecommunication Systems*, 78(4) :559–576, 2021.

[71] Chandra Naik and D Pushparaj Shetty. Optimal sensors placement scheme for targets coverage with minimized interference using bbo. *Evolutionary Intelligence*, pages 1–15, 2021.

[72] Mudassar Ali Syed and Raziuddin Syed. Weighted salp swarm algorithm and its applications towards optimal sensor deployment. *Journal of King Saud University-Computer and Information Sciences*, 34(4) :1285–1295, 2022.

[73] A Nageswar Rao, B Rajendra Naik, and L Nirmala Devi. On the relay node placement in wsns for lifetime maximization through metaheuristics. *Materials Today : Proceedings*, 2020.

[74] Slaheddine Chelbi, Habib Dhahri, and Rafik Bouaziz. Node placement optimization using particle swarm optimization and iterated local search algorithm in wireless sensor networks. *International Journal of Communication Systems*, 34(9) :e4813, 2021.

[75] Arouna Ndam Njoya, Ado Adamou Abba Ari, Marah Nana Awa, Chafiq Titouna, Nabila Labraoui, Joseph Yves Effa, Wahabou Abdou, and Abdelhak Gueroui. Hybrid wireless sensors deployment scheme with connectivity and coverage maintaining in wireless sensor networks. *Wireless Personal Communications*, 112(3) :1893–1917, 2020.

[76] Yahui Sun and Saman Halgamuge. Minimum-cost heterogeneous node placement in wireless sensor networks. *IEEE Access*, 7 :14847–14858, 2019.

[77] Slimane Charafeddine Benghelima, Mohamed Ould-Khaoua, Ali Benzerbadj, and Oumaya Baala. Multi-objective optimisation of wireless sensor networks deployment : Application to fire surveillance in smart car parks. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pages 98–104, 2021.

[78] Saunhita Sapre and S Mini. Moth flame optimization algorithm based on decomposition for placement of relay nodes in wsns. *Wireless Networks*, 26(2) :1473–1492, 2020.

[79] Gianni A Di Caro and Eduardo Feo Flushing. Optimal relay node placement for throughput enhancement in wireless sensor networks. In *2011 50th FITCE Congress-" ICT : Bridging an Ever Shifting Digital Divide"*, pages 1–6. IEEE, 2011.

[80] A. H. Alavi, Pengcheng Jiao, W. Buttlar, and N. Lajnef. Internet of things-enabled smart cities : State-of-the-art and future trends. *Measurement*, 129, 2018.

[81] Qianqian Yu, Chen Yang, Guangming Dai, Lei Peng, and Xiaoyu Chen. Synchronous wireless sensor and sink placement method using dual-population co-evolutionary constrained multiobjective optimization algorithm. *IEEE Transactions on Industrial Informatics*, 19(6) :7561–7571, 2023.

[82] Performance analysis of proximity and light sensors for smart parking. *Procedia Computer Science*, 83 :385–392, 2016.

[83] M. Y. I. Idris, Emran Mohd Tamil, Zaidi Razak, Noorzaily Mohamed Noor, and L. Kin. Smart parking system using image processing techniques in wireless sensor network environment. *Information Technology Journal*, 8 :114–127, 2009.

[84] Steve Azarsh Ratti, Nasrullah Pirzada, Syed Mohammad Ali Shah, and Anjum Naveed. Intelligent car parking system using wsn. In *2023 Global Conference on Wireless and Optical Technologies (GCWOT)*, pages 1–9, 2023.

[85] Xiuming Guo, Chunjiang Zhao, Xinting Yang, and Chuanheng Sun. A deterministic sensor node deployment method with target coverage and node connectivity. In Hepu Deng, Duoqian Miao, Jingsheng Lei, and Fu Lee Wang, editors, *Artificial Intelligence and Computational Intelligence*, pages 201–207. Springer Berlin Heidelberg, 2011.

[86] S. Abdollahzadeh and Nima Jafari Navimipour. Deployment strategies in the wireless sensor network : A comprehensive review. *Comput. Commun.*, 91-92 :1–16, 2016.

[87] J. L. Bredin, E. D. Demaine, M. T. Hajiaghayi, and D. Rus. Deploying sensor networks with guaranteed fault tolerance. *IEEE/ACM Transactions on Networking*, 18(1) :216–228, 2010.

[88] Y. Sun, D. Rehfeldt, M. Brazil, D. Thomas, and S. Halgamuge. A physarum-inspired algorithm for minimum-cost relay node placement in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 28(2) :681–694, 2020.

[89] Hakim Q. A. Abdulrab, Fawnizu Azmadi Hussin, Idris Ismail, Maher Assaad, Azlan Awang, Hussein Shutari, and P. Arun Mozhi Devan. Hybrid harris hawks with sine cosine for optimal node placement and congestion reduction in an industrial wireless mesh network. *IEEE Access*, 11 :2500–2523, 2023.

[90] Lixin Wang, Peng-Jun Wan, and William Washington. Connectivity of multihop wireless networks with log-normal shadowing. *Wireless Networks*, 21(7) :2279–2292, 2015.

[91] Miloud Bagaa, Ali Chelli, Djamel Djenouri, Tarik Taleb, Ilangko Balasingham, and Kimmo Kansanen. Optimal placement of relay nodes over limited positions in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 16(4) :2205–2219, 2017.

[92] Ricardo Landa Becerra and Carlos A. Coello Coello. Solving hard multiobjective optimization problems using ε-constraint with cultured differential evolution. Springer Berlin Heidelberg, 2006.

[93] Arouna Ndam Njoya, Ado Adamou Abba Ari, Marah Nana Awa, Chafiq Titouna, Nabila Labraoui, Joseph Yves Effa, Wahabou Abdou, and Abdelhak Gueroui. Hybrid Wireless Sensors Deployment Scheme with Connectivity and Coverage Maintaining in Wireless Sensor Networks. *Wireless Personal Communications*, 2020.

[94] Nguyen Thi Hanh, Huynh Thi Thanh Binh, Nguyen Van Son, and Phan Ngoc Lan. Minimal Node Placement for Ensuring Target Coverage With Network Connectivity and Fault Tolerance Constraints in Wireless Sensor Networks. In *2019 IEEE Congress on Evolutionary Computation (CEC)*.

[95] Y. Sun and S. Halgamuge. Minimum-cost heterogeneous node placement in wireless sensor networks. *IEEE Access*, 7 :14847–14858, 2019.

[96] Gurobi Optimization. Inc.,"gurobi optimizer reference manual," 2015.

[97] Muhammad Zia Ul Haq, Muhammad Zahid Khan, Haseeb Ur Rehman, Gulzar Mehmood, Ahmed Binmahfoudh, Moez Krichen, and Roobaea Alroobaea. An adaptive topology management scheme to maintain network connectivity in wireless sensor networks. *Sensors*, 22(8) :2855, 2022.

[98] Mohammad Abdul Matin and MM Islam. Overview of wireless sensor network. *Wireless sensor networks-technology and protocols*, pages 1–3, 2012.

[99] Ahmed M. Khedr, S. Sheeja Rani, and Mohamed Saad. Optimized deep learning for congestion-aware continuous target tracking and boundary detection in iot-assisted wsn. *IEEE Sensors Journal*, 23(7) :7938–7948, 2023.

[100] Mohammed Farsi, Mostafa A. Elhosseini, Mahmoud Badawy, Hesham Arafat Ali, and Hanaa Zain Eldin. Deployment techniques in wireless sensor networks, coverage and connectivity : A survey. *IEEE Access*, 7 :28940–28954, 2019.

[101] Wei Liang, Chaofan Ma, Meng Zheng, and Longxiang Luo. Relay node placement in wireless sensor networks : from theory to practice. *IEEE Transactions on Mobile Computing*, 20(4) :1602–1613, 2019.

[102] R Timothy Marler and Jasbir S Arora. The weighted sum method for multi-objective optimization : new insights. *Structural and multidisciplinary optimization*, 41(6) :853–862, 2010.

[103] Chaofan Ma, Wei Liang, Meng Zheng, and Bo Yang. Relay node placement in wireless sensor networks with respect to delay and reliability requirements. *IEEE Systems Journal*, 13(3) :2570–2581, 2018.

[104] Xuxun Liu. A deployment strategy for multiple types of requirements in wireless sensor networks. *IEEE Transactions on Cybernetics*, 45(10) :2364–2376, 2015.

[105] Azzedine Boukerche and Peng Sun. Connectivity and coverage based protocols for wireless sensor networks. *Ad Hoc Networks*, 80 :54–69, 2018.

[106] Chaofan Ma, Wei Liang, and Meng Zheng. Delay constrained relay node placement in two-tiered wireless sensor networks : A set-covering-based algorithm. *Journal of Network and Computer Applications*, 93 :76–90, 2017.

[107] Maher Rebai, Matthieu Le Berre, Hichem Snoussi, Faicel Hnaien, and Lyes Khoukhi. Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks. *Computers and Operations Research*, 59 :11–21, July 2015.

[108] Ricardo Landa Becerra and Carlos A Coello Coello. Solving hard multiobjective optimization problems using ε-constraint with cultured differential evolution. In *Parallel Problem Solving from Nature-PPSN IX*, pages 543–552. Springer, 2006.

[109] Seyedali Mirjalili and Andrew Lewis. The whale optimization algorithm. *Advances in engineering software*, 95 :51–67, 2016.

[110] Slimane Charafeddine Benghelima, Mohamed Ould-Khaoua, Ali Benzerbadj, Oumaya Baala, and Jalel Ben-Othman. Optimization of the deployment of wireless sensor networks dedicated to fire detection in smart car parks using chaos whale optimization algorithm. In *ICC 2022-IEEE International Conference on Communications*, pages 3592–3597. IEEE, 2022.

[111] Harun Bingol and Bilal Alatas. Chaotic league championship algorithms. *Arabian journal for science and engineering*, 41 :5123–5147, 2016.

[112] Elif Varol Altay and Bilal Alatas. Bird swarm algorithms with chaotic mapping. *Artificial Intelligence Review*, 53 :1373–1414, 2020.

[113] Rui Tang, Simon Fong, and Nilanjan Dey. Metaheuristics and chaos theory. *Chaos theory*, pages 182–196, 2018.

[114] R Timothy Marler and Jasbir S Arora. Function-transformation methods for multi-objective optimization. *Engineering Optimization*, 37(6) :551–570, 2005.

[115] Alan Bundy and Lincoln Wallen. Breadth-first search. In *Catalogue of artificial intelligence tools*, pages 13–13. Springer, 1984.

[116] R Deepa and Revathi Venkataraman. Enhancing whale optimization algorithm with levy flight for coverage optimization in wireless sensor networks. *Computers & Electrical Engineering*, 94 :107359, 2021.

[117] Mrutyunjay Rout and Rajarshi Roy. Optimal wireless sensor network information coverage using particle swarm optimisation method. *International Journal of Electronics Letters*, 5(4) :491–499, 2017.