الجمهوريـــة الجزائريـــة الديمقراطيـــة الشعبيــة

People's Democratic Republic of Algeria

وزارة الـتعلــيـم الـعـالي و الـبحـث الـعلـمــي

Ministry of Higher Education and Scientific Research

MUSTAPHA Stambouli University
Mascara

جامعة مصطفى اسطمبولي
معسكر

Faculty of Exact Sciences

Department of Computer Science

DOCTORAL THESIS

Speciality : Computer science
Option : Information and Communication Technology

Entitled

Bio-inspired algorithms for optimized deployment of sensor nodes in wireless networks

Presented by : DEGHBOUCH Hicham

On the 15/06/2022

In front of the jury :

Chairman	SMAIL Omar	MCA	University of Mascara
Examiner	LOUAZANI Ahmed	MCA	University of Relizane
Examiner	BOUFERA Fatma	MCA	University of Mascara
Examiner	TEGGAR Hamza	MCA	University of Mascara
Supervisor	DEBBAT Fatima	Professor	University of Mascara

Academic Year : 2021-2022

الجمهوريـــة الجزائريــة الديمقراطيــة الشعبيــة

République Algérienne Démocratique et Populaire

وزارة الـتعـلـيـم الـعـالي و البـحـث الـعـلـمـي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université MUSTAPHA Stambouli
Mascara

جامعة مصطفى اسطمبولي
معسكر

Faculté des Sciences Exactes

Département d'Informatique

THESE de DOCTORAT de 3ème cycle

Spécialité : Informatique

Option : Technologie de l'information et de la communication

Intitulée

Algorithmes bio-inspirés pour le déploiement optimisé des nœuds de capteurs dans les réseaux sans fil

Présentée par : DEGHBOUCH Hicham

Le 15/06/2022

Devant le jury :

Président	SMAIL Omar	MCA	Université de Mascara
Examinateur	LOUAZANI Ahmed	MCA	Université de Relizane
Examinateur	BOUFERA Fatma	MCA	Université de Mascara
Examinateur	TEGGAR Hamza	MCA	Université de Mascara
Encadreur	DEBBAT Fatima	Professeur	Université de Mascara

Année Universitaire : 2021-2022

ACKNOWLEDGEMENTS

First and foremost, I want to say Alhamdulillah, I thank Allah for giving me the strength, the courage, and the eagerness to complete this thesis.

I would like to express my sincere gratitude to my supervisor Pr. Debbat Fatima for her consistent support, encouragement, and guidance during the running of this project. Her thoughtful comments and recommendations were vital in inspiring me to think outside the box.

In addition, I want to thank the members of my Ph.D. committee, professors SMAIL Omar, LOUAZANI Ahmed, BOUFERA Fatma, and TEGGAR Hamza for their interest in my work and their valuable evaluations.

To conclude, I cannot forget to thank my family and friends for all the unconditional support in these very intense academic years.

DEDICATION

To my dear parents,
To my lovely wife,
My brothers and my sister.

RÉSUMÉ DÉTAILLÉ

Un réseau de capteurs sans fil (RCSF) est un type particulier de réseau ad-hoc composé d'un ensemble de capteurs allant de quelques-uns à des milliers. Ces capteurs sont de petits appareils avec des ressources de traitement et d'alimentation limitées. Ils s'appuient sur des batteries limitées et parfois immuable pour effectuer leurs tâches. Les capteurs sont répartis dans une région géographique et organisés en un réseau coopératif pour effectuer la tâche de surveillance.

Les réseaux de capteurs sans fil (RCSFs) sont des solutions rentables car ils n'utilisent aucun type de liaisons filaires entre les nœuds. De plus, l'absence de liaisons filaires donne aux nœuds une liberté de mouvement sans perdre la connectivité. En outre, les RCSFs sont des réseaux flexibles caractérisés par l'intégration rapide et facile de nouveaux nœuds. Toutes ces fonctionnalités et bien plus font des RCSFs le premier choix pour plusieurs applications de surveillance. Les RCSFs ont la capacité de fournir une surveillance fiable pour de nombreuses applications du monde réel telles que la surveillance du trafic, les soins de santé, les diagnostics de panne de machine, la surveillance environnementale..., etc.

Malgré les nombreux avantages des réseaux de capteurs sans fil qui les rendent dominants dans la surveillance à distance, plusieurs problèmes qui affectent la capacité de détection du réseau doivent être résolus lors de leur mise en œuvre, tels que le déploiement des nœuds de capteurs.

Le problème de déploiement des RCSF est une étape essentielle dans la construction d'un réseau robuste capable de satisfaire les exigences applicatives. L'objectif de la stratégie de déploiement est de définir la topologie du réseau en plaçant les capteurs à des emplacements optimaux. Les capteurs sont placés de manière à satisfaire une ou plusieurs propriétés intrinsèques du RCSF telles que la couverture, la connectivité, le coût et la durée de vie du réseau. De plus, en utilisant le plan de déploiement approprié, la complexité de plusieurs problèmes dans les réseaux de capteurs sans fil peut être réduite, tels que le routage et la conservation de l'énergie. Par conséquent, nous pouvons dire que le bon fonctionnement du RCSF dépend fortement de la position des capteurs.

La stratégie de déploiement est déterminée en fonction de l'application et de la nature de l'environnement. Lors du déploiement du RCSF dans un environnement convivial, les capteurs

sont placés à des coordonnées précises en s'appuyant sur un modèle déterministe prédéfini. Cependant, dans les environnements inaccessibles ou risqués, le seul choix disponible est le déploiement aléatoire, dans lequel les capteurs sont dispersés depuis l'air sur la zone de détection. En clair, lorsque les capteurs sont déployés au hasard, ils n'assureront pas les performances attendues, car il y a de fortes chances qu'ils se retrouvent placés les uns au-dessus des autres là où leurs zones de couverture se chevauchent. Il en résulte une mauvaise couverture empêchant le RCSF d'atteindre les objectifs souhaités. Dans le cas d'un déploiement aléatoire, la plupart des applications utilisent des capteurs mobiles, ce qui leur donne la possibilité d'améliorer la couverture en déplaçant les capteurs vers de nouveaux emplacements après la diffusion aléatoire initiale.

Dans ce travail de thèse, nous nous intéressons à la résolution du problème de déploiement par l'utilisation des algorithmes bio-inspirés. La popularité de ces algorithmes est due à de nombreuses raisons, telles que leur efficacité et leur faible complexité de calcul par rapport aux autres techniques d'optimisation déterministes existantes. De plus, ils ont la capacité d'accélérer le processus de recherche d'un optimum global ou d'une solution approchée de l'optimum pour les cas où les méthodes déterministes ne parviennent pas à trouver rapidement une solution satisfaisante.

Dans ce contexte, nous avons proposé deux approches de déploiement pour placer de manière optimale les nœuds capteurs dans une zone d'intérêt. Les deux techniques de déploiement sont basées sur un algorithme bio-inspiré populaire appelé Bees Algorithm (BA). C'est un algorithme qui imite le comportement des abeilles lorsqu'elles cherchent de la nourriture. Il est basé sur les interactions entre les abeilles d'une colonie lors de la récolte du nectar.

La première contribution concerne le déploiement de RCSF homogènes et hétérogènes en maximisant la zone couverte après un déploiement aléatoire initial. Pour atteindre cet objectif, nous commençons par représenter la position de chaque source de nourriture, qui est considérée comme une solution dans le BA par un vecteur qui contient toutes les coordonnées de position des capteurs. Après la représentation de la solution, plutôt que d'utiliser les étapes du BA d'origine qui souffrent de plusieurs problèmes, notamment la stagnation de la recherche et la recherche aléatoire non guidée, nous avons introduit un nouvel algorithme appelé Improved Bees Algorithm (IBA).

Le BA dans sa forme de base effectue une recherche de voisinage exploitante dans les régions les plus prometteuses dans l'espoir de trouver des solutions plus précises. Cependant, pour atteindre la précision de recherche souhaitée, la recherche locale doit être concentrée autour des meilleures solutions en rétrécissant l'espace de recherche local. Ce mécanisme augmente les chances d'obtenir une bonne approximation de l'optimum global. De plus, éviter la stagnation de l'optimum local est une exigence majeure en particulier lorsque la recherche locale dans un champ de fleurs ne parvient pas à améliorer la qualité des solutions produites. A cet égard, deux procédures nommées "neighborhood shrinking" et "site abandonment" sont introduites dans le BA.

Les deux améliorations apportées au BA de base renforcent l'algorithme en améliorant l'efficacité de la recherche et permettent une exploitation plus précise en rétrécissant l'espace de recherche local. Le but de la procédure de rétrécissement est de garder la recherche concentrée au-

tour des meilleures solutions dans l'espoir de trouver des positions optimales de capteurs qui améliorent la couverture du RCSF. De plus, la procédure d'abandon de site renforce la capacité de l'algorithme à sortir de l'optimum local lorsqu'il n'y a pas d'amélioration. Les performances de l'IBA sont évaluées à l'aide d'un nombre et d'un type de capteurs variés. Les études comparatives menées montrent que l'IBA offre de bons résultats de couverture et de bons schémas de déploiement dans le cas de RCSF homogènes et hétérogènes.
La deuxième contribution est une version étendue de la première, dans laquelle la consommation d'énergie lors des mouvements du capteur est considérée. Les capteurs doivent passer de leurs positions aléatoires initiales aux positions finales optimisées par l'algorithme d'optimisation. Par conséquent, les emplacements finaux des capteurs doivent être choisis avec soin afin de réduire la distance de déplacement des capteurs, ce qui en retour réduit la quantité d'énergie consommée lors du déplacement des capteurs. Pour atteindre cet objectif, nous avons proposé une nouvelle technique de déploiement qui prend en compte la couverture et la durée de vie du réseau lors de la planification du déploiement des capteurs. La technique proposée est développée en hybridant deux algorithmes bio-inspirés à savoir le Bees Algorithm (BA) et le Grasshopper Optimization Algorithm (GOA).
De nombreux algorithmes bio-inspirés tels que le BA rencontrent des difficultés lors de l'optimisation d'un certain nombre de problèmes. Par exemple, lorsque la dimension du problème est élevée (comme dans le cas du déploiement de RCSF), ils échouent à produire des solutions de haute qualité, ce qui les empêche d'atteindre des solutions approximatives de l'optimum global. De plus, lorsque l'algorithme repose sur un degré plus élevé d'aléatoire comme BA, les agents de recherche subiront des mouvements imprévisibles et à longue distance pendant la recherche en raison du comportement aléatoire. Ce type de mouvement augmentera la distance de déplacement des capteurs, ce qui augmentera la quantité d'énergie épuisée lors du déplacement des capteurs.
Pour pallier ces lacunes, nous avons intégré l'algorithme GOA en tant qu'opérateur dans la BA afin d'améliorer l'exploitation et la précision de la recherche locale. La force de GOA réside dans son haut niveau d'exploitation guidé par les interactions sociales entre tous les agents de l'essaim, ce qui le rend idéal pour l'hybridation avec le BA. En hybridant les deux algorithmes, notre stratégie a montré une amélioration remarquable en termes de couverture et de consommation d'énergie et a prouvé qu'elle avait une excellente adaptabilité pour résoudre différents problèmes de déploiement.
La qualité de nos solutions est confirmée par les performances élevées enregistrées lors de la comparaison avec les solutions récemment proposées dans la littérature. Les résultats présentés montrent que les approches proposées offrent des performances significativement meilleures dans tous les cas de test en termes de qualité de solution, de convergence et de stabilité, même lorsque le réseau est composé à la fois de capteurs mobiles et fixes.

ABSTRACT

Wireless Sensor Networks (WSNs) are a type of ad-hoc network technology that has been around for more than two decades. WSNs typically consist of a number of dedicated sensors, which are fundamentally low-cost, autonomous, resource-constrained devices organized into a cooperative network to perform a common monitoring task. From their first appearance until the present day, there has been a significant effort conducted by researchers to achieve a reliable WSN design that can provide better Quality of Service (QoS) for a wide range of applications. Deployment optimization is one of the crucial issues that must be taken into consideration while designing an efficient WSN. What is meant by deployment optimization is that the sensors must be placed in strategic locations that optimize one or multiple design criteria including, coverage, connectivity, cost, and network lifetime.

In this thesis, we have addressed the problem of deployment by using bio-inspired algorithms. We proposed two deployment solutions for optimally placing homogeneous and heterogeneous WSNs. The proposed bio-inspired solutions allow the relocation of network sensors to locations that optimize coverage and energy efficiency. The first solution IBA achieves the desired objectives by eliminating both coverage redundancy and coverage holes resulted after the random deployment of heterogeneous sensors. Whereas the second solution BAGOA, which is developed by hybridizing two algorithms, namely the Bees Algorithm (BA) and the Grasshopper Optimization Algorithm (GOA), achieves high coverage and ensures low mobility during deployment in several deployment situations, even when the network is composed of both mobile and stationary sensors. The effectiveness of the proposed solutions is confirmed by the high performance recorded during the comparison with recently proposed solutions in the literature. The presented results show that IBA and BAGOA provide significantly better performance in all deployment test cases in terms of solution quality, convergence, and stability.

Keywords: WSN, Sensor deployment, Optimization, Bio-inspired algorithms, BA, GOA.

RÉSUMÉ

Les réseaux de capteurs sans fil (RCSFs) sont un type de technologie de réseau ad hoc qui existe depuis plus de deux décennies. Les RCSFs se composent généralement d'un certain nombre de capteurs dédiés, qui sont des dispositifs fondamentalement peu coûteux, autonomes et limités en ressources, organisés en un réseau coopératif pour effectuer une tâche de surveillance commune. Depuis leur première apparition jusqu'à nos jours, des efforts importants ont été déployés par les chercheurs pour parvenir à une conception fiable qui peut fournir une meilleure qualité de service (QoS) pour un large éventail d'applications. L'optimisation du déploiement est l'un des enjeux cruciaux qui doit être pris en considération lors de la conception d'un RCSF efficace. L'optimisation du déploiement signifie que les capteurs doivent être placés à des emplacements stratégiques qui optimisent un ou plusieurs critères de conception, notamment la couverture, la connectivité, le coût et la durée de vie du réseau.

Dans cette thèse, nous avons traité le problème du déploiement en utilisant des algorithmes bio-inspirés. Nous avons proposé deux solutions de déploiement pour placer de manière optimale des RCSFs homogènes et hétérogènes. Les solutions bio-inspirées proposées permettent de déplacer les capteurs du réseau vers des emplacements qui optimisent la couverture et efficacité énergétique. La première solution IBA atteint les objectifs souhaités en éliminant à la fois la redondance de couverture et les trous de couverture résultant du déploiement aléatoire de capteurs hétérogènes. Alors que la seconde solution BAGOA, qui est développée en hybridant deux algorithmes, à savoir Bees Algorithm (BA) et Grasshopper Optimization Algorithm (GOA), atteint une couverture élevée et assure une faible mobilité pendant le déploiement dans plusieurs situations, même lorsque le réseau est composé à la fois de capteurs mobiles et fixes. L'efficacité des solutions proposées est confirmée par la performance élevée enregistrée lors de la comparaison avec les solutions récemment proposées dans la littérature. Les résultats présentés montrent qu'IBA et BAGOA offrent des performances nettement meilleures dans tous les cas de test de déploiement en termes de qualité de solution, de convergence et de stabilité.

Mots clés: RCSF, Déploiement des capteurs, Optimisation, Algorithmes Bio-inspirés, BA, GOA.

ملخص

شبكات الاستشعار اللاسلكية هي نوع من تقنيات الشبكات Ad hoc التي تتواجد منذ أكثر من عقدين. تتكون شبكات الاستشعار اللاسلكية عادةً من عدد من أجهزة الاستشعار المخصصة، وهي أجهزة منخفضة التكلفة ومستقلة ومحدودة الموارد تكون منظمة في شبكة تعاونية من أجل أداء مهمة مراقبة مشتركة. منذ ظهورها لأول مرة حتى يومنا هذا، هناك جهد كبير قام به الباحثون لتحقيق تصميم موثوق لشبكات الاستشعار اللاسلكية الذي يمكنه أن يوفر جودة خدمة أفضل لمجموعة واسعة من التطبيقات. يعد تحسين النشر أحد المشكلات الحاسمة التي يجب أخذها في الاعتبار أثناء تصميم شبكة استشعار لاسلكية فعالة. المقصود بتحسين النشر هو أنه يجب وضع المستشعرات في مواقع استراتيجية تعمل على تحسين معيار تصميم واحد أو عدة معايير مثل التغطية والاتصال والتكلفة وعمر الشبكة.

في هذه الأطروحة، عالجنا مشكلة النشر باستخدام خوارزميات مستوحاة من البيولوجيا. لقد اقترحنا حلين للنشر من أجل وضع شبكات الاستشعار اللاسلكية المتجانسة وغير المتجانسة على النحو الأمثل. تسمح الحلول المقترحة المستوحاة من الحيوية بنقل مستشعرات الشبكة إلى المواقع التي تعمل على تحسين التغطية وكفاءة الطاقة. يحقق الحل الأول IBA الأهداف المرجوة من خلال القضاء على كل من تكرار التغطية وثغرات التغطية الناتجة بعد النشر العشوائي لأجهزة الاستشعار غير المتجانسة. في حين أن الحل الثاني BAGOA، الذي تم تطويره من خلال تهجين خوارزميتين، هما خوارزمية النحل و خوارزمية الجندب، يحقق تغطية عالية ويضمن تنقلاً منخفضاً أثناء النشر في العديد من الحالات، حتى عندما تكون الشبكة متكونة من مستشعرات متنقلة وثابتة. تم تأكيد فعالية الحلول المقترحة من خلال الأداء العالي المسجل أثناء المقارنة مع الحلول المقترحة مؤخراً في الأدبيات. تظهر النتائج المقدمة أن حلينا يوفران أداءً أفضل بشكل ملحوظ في جميع حالات اختبار النشر من حيث جودة الحل والتقارب والاستقرار.

الكلمات الرئيسية: شبكات الاستشعار اللاسلكية، نشر أجهزة الاستشعار، التحسين، المستوحاة من البيولوجيا ، BA، GOA.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Context

In recent years, there has been a worldwide interest in developing the so-called "next-generation monitoring and control systems". These systems are supposed to provide rapid data collection with a minimum price. The continuous demand for such systems in several domains has attracted the attention of researchers and industrial communities around the globe. This interest triggers a remarkable technological development in the last few decades, especially in Micro-Electro-Mechanical-Systems (MEMS) and wireless communications, which led to the emergence of a successful solution in data monitoring called the sensor.
The proliferation in MEMS technology has facilitated the development of smart sensors with a reduced size that has the capability to monitor large regions over the ground surface, underwater, or in the atmosphere. Sensors are small devices with limited processing and power resources. They rely on limited and sometimes unchangeable batteries to perform their tasks. Due to these limitations, the sensor as an autonomous entity is not capable on its own to perform all the monitoring tasks such as sensing, transmitting, and relaying data, especially in large geographical regions. Therefore, for efficient monitoring, it is feasible to deploy large numbers of sensors that are organized in networks, in which they sense the surrounding environment and cooperate in order to communicate the gathered information to the end-user through wireless links. This type of cooperative network is known as the Wireless Sensor Network (WSN).
The origin of WSNs can be seen in military and heavy industrial applications. Specifically, the first designed wireless network that resembles a modern WSN is the Sound Surveillance System (SOSUS). It was developed by the United States Military in the 1950s to detect and track Soviet submarines. SOSUS used submerged acoustic sensors distributed in the Atlantic and Pacific oceans. Today, recent advances have combined to enable a new generation of WSNs that differ greatly from wireless networks developed in the 1950s. Modern WSNs are becoming involved in almost every field of life. Many real-world applications such as traffic surveillance, security monitoring, health care, machine failure diagnoses, and environmental monitoring benefit from the services provided by these networks.

WSNs offer numerous advantages that make them dominant in remote monitoring, however, several design problems must be addressed while implementing them. One important problem that significantly affects many intrinsic performance criterions such as coverage, connectivity, cost, and network lifetime, is the deployment problem. The deployment of wireless sensor networks is an essential step in constructing a robust network that is capable of satisfying the design specifications. The aim of the deployment strategy is to define the topology of the network i.e the number of sensors and their optimal locations. The word optimal here means the choice of the most appropriate locations for sensors from a set of finite potential locations that satisfy the user requirements. Therefore, the task of the deployment strategy is not a simple choice of locations among many, but the "best" choice of locations to place the sensors taking into consideration all user requirements.
The optimal deployment of WSNs was defined as an NP-hard optimization problem in most works in the literature. The complexity of the deployment problem arises from several aspects, such as the constrained nature of sensors and the environmental conditions. When deploying the WSN in a friendly environment, the sensors are placed at accurate coordinates using the hands or robots relying on a predefined deterministic pattern. However, in unreachable or risky environments, the deployment becomes a challenging task that requires tight planning. Besides, optimizing conflicting objectives such as maximizing coverage and minimizing energy consumption adds an additional layer of complexity to the deployment problem, which makes deployment optimization a very critical issue. Exact resolution techniques are not preferable when solving these types of problems where the computational time taken by the algorithms increases exponentially with the problem dimension. As an alternative, metaheuristics, in particular, bio-inspired algorithms have been used for obtaining the optimal solutions of various engineering design optimization problems.

Objectives

The main objective of this thesis is to solve the deployment problem of wireless sensor networks using bio-inspired algorithms. In our research, we have considered the deployment of homogeneous WSNs where several sensors with the same characteristics are deployed. Furthermore, we have also considered the deployment of heterogeneous WSNs where multiple sensor types with different characteristics are deployed. In the heterogeneous case, the sensors not only differ in the sensing and communication ranges but also differ in the type of sensors where both stationary and mobile sensors are considered. In this case, the deployment is much complex compared with the deployment of homogeneous sensors because several problems arise, such as coverage redundancy.
To achieve the aforementioned objectives, we propose several techniques based on bio-inspired algorithms that aim to achieve the optimal deployment, or at least the best approximation of the optimal. The ultimate goal of our research can be summarized as follows:

- Maximize the coverage of the targets in the predefined sensing area after initial random deployment.

- Ensure the minimum energy consumption through reducing the moving distance of sensors during displacement from their initial random positions to their final positions optimized by the algorithms.

Contributions

This thesis has been the subject of two contributions that we can summarize as follows:
The first contribution allows addressing the deployment of both homogeneous and heterogeneous WSNs through maximizing the covered area. We have proposed a novel deployment strategy for maximizing the area coverage after initial random scattering of sensors. Firstly, we have given a mathematical formulation to measure the quality of coverage of targets as a function of the distances between them and the sensors. Secondly, we have designed our strategy, which is based on a bio-inspired algorithm, to provide optimal deployment patterns regardless of the number of deployed sensors and their initial random locations. The proposed strategy ensures high coverage quality and continuous monitoring of the area of interest through eliminating coverage holes and coverage redundancy, especially in heterogeneous mobile WSNs. Moreover, we designed our strategy to guarantee the choice of the appropriate positions for the sensors despite their sensing ranges in a reasonable time. By achieving these objectives, our strategy can be regarded as an efficient tool that helps end-users and system deciders in decision making by collecting useful and accurate data.

In the second contribution, in addition to coverage, we were interested in the problem of energy consumption. There are several techniques used for conserving the energy of sensors while the WSN is operational. However, in order to save sensors' energy during deployment, the movement of sensors must be limited and strategically planned because the big portion of sensors' energy is depleted during movement. In our contribution, we present a solution for reducing energy consumption during deployment through minimizing the moving distance of sensors. For this purpose, we have designed a novel strategy by hybridizing two bio-inspired algorithms for solving several deployment problems. We have started by solving deployment problems with small sensing areas where several sensor densities are used. We next moved on to solving deployment problems with large instances in larger sensing areas. Finally, we passed on solving deployment problems of a mixed WSN, in which several mobile sensors are deployed optimally to enhance the coverage of stationary sensors.

Manuscript organization

This thesis is organized as follows:
In chapter I, we begin by giving a general overview of wireless sensor networks. In this overview, we highlight the different key elements of these networks, and we describe their various functions.

In chapter II, we focus on giving a detailed overview of combinatorial optimization problems and their resolution techniques, namely, stochastic optimization. We will further present the different classes of stochastic optimization, including heuristics, metaheuristics, nature-inspired algorithms, and bio-inspired algorithms.

Chapter III aims to introduce the deployment problem in wireless sensor networks and highlight the most important objectives, which have a great impact on the performance of WSNs. Besides, it provides a classification of the proposed deployment solutions with their advantages and disadvantages.

Chapter IV presents our first contribution that aims to solve the deployment problem of homogeneous and heterogeneous wireless sensor networks by optimizing the coverage.

Chapter V presents our second contribution to the deployment problem. In this chapter, our objective is to maximize the coverage and minimize the energy consumption during the displacement of sensors. We start by presenting our new solution, and then we conduct a set of comparative studies using several deployment situations and settings.

The thesis ends with a general conclusion and some potential future works.

CHAPTER I

WIRELESS SENSOR NETWORKS

Contents

I.1 Introduction

Wireless networks have become an ubiquitous technology used by various applications around the developed world. These networks offer a huge amount of services to people who exploit them to improve their lives and works. Among these networks, Ad hoc wireless networks have emerged as a successful solution that penetrates the development market day after day through introducing new advancements in many fields. A wireless ad hoc network is an assemblage of devices called nodes, which cooperatively interconnect to form a functional network. The nodes may be located in/on various things such as airplanes, ships, trucks, cars, or even on people. The nodes are equipped with wireless antennas that support omnidirectional (broadcast) and highly directional (point-to-point) communication for transmitting and receiving signals. The main characteristic of an ad hoc network is that the nodes are capable to communicate freely without any fixed infrastructure or centralized control [1]. Each node is, therefore, plays the role of an access point and a host simultaneously. The wireless ad hoc networks are differentiated from the other networks by the use of wireless links. The nodes used to form an ad hoc network are generally constrained in terms of resources such as the transmission range, which prevents them from communicating directly with each other. Therefore, the nodes are required to collaborate in order to forward the information from the source to the destination [2]. This can be achieved by ensuring that every two nodes have at least one complete path between them directly or through other nodes in the network. Thus, the use of multi-hop routes is a basic requirement in ad hoc networks for relaying packets towards their final destination [3, 4].

There are several types of ad hoc networks, each of which is designed to serve the needs of a number of applications. The most popular type is Mobile Ad Hoc Network (MANET). MANET is a gathering of autonomous mobile wireless nodes that are connected via multiple wireless links. The nodes are capable to communicate in the absence of a fixed infrastructure. MANET can be operated as a stand-alone network or may be connected to a large network such as the internet. This property makes MANET a very attractive solution to various applications [5, 6]. Another frequently used network is called wireless mesh ad hoc network. A mesh network is a multi-hop wireless network formed by a number of wireless nodes called mesh nodes or mesh routers. The interconnection between them forms an ad hoc backbone structure for the mesh network. Besides, some mesh nodes function as a wireless access point through which the wireless clients attach themselves to the network. The main use of a wireless mesh network is to extend the network coverage to support a greater number of nodes and maintain the required end-to-end connectivity [7].

In addition to the two types presented above, there is another type of ad hoc network that groups all the aforementioned characteristics. This special type is known as the Wireless Sensor Network (WSN). WSN is considered a very successful solution employed by countless applications that make use of its services in remote surveillance and data gathering. The rest of this chapter will focus on giving a detailed overview of the wireless sensor networks and their architecture, followed by a comprehensive review of their types, standards, and operating systems. In the last two sections, we highlight the popular applications of WSNs as well as the different design problems encountered during the implementation of these networks.

I.2 Wireless Sensor Networks (WSNs)

In the age of information, the continuous demand for powerful systems that can provide rapid data collection with the minimum price has attracted the attention of both academic and industrial communities. This interest triggers remarkable technological development in electronics and wireless telecommunications that has paved the way for the development of a successful solution in data gathering, a small-sized electronic device called the sensor [8]. The sensor is an autonomous entity associated with a battery as its energy source. Moreover, it has the capability to sense the surroundings and collect useful data about the sensed phenomenon [9]. These sensors are used in various applications to perform different tasks, such as smart monitoring, data processing, data storage, data gathering, target tracking, and controlling.
A Wireless Sensor Network (WSN) is a particular type of ad hoc network that is composed of a collection of sensors that varies from a few to thousands. These sensors are distributed in a geographical region and organized into a cooperative network to perform the monitoring task. Multiple environmental conditions can be monitored by WSNs including, sound, wind, pressure, and humidity. Besides, WSNs have the capability to provide reliable monitoring for many real-world applications such as traffic surveillance, security monitoring, health care, machine failure diagnoses, and environmental monitoring. Sensors keep track of the events that take place in their surroundings and communicate with each other to forward the gathered data to the sink for processing [10]. After the recipient of the gathered data, the sink either uses the data locally or forwards it through a gateway to other networks such as the Internet. Figure I.1 shows the general architecture of WSNs.

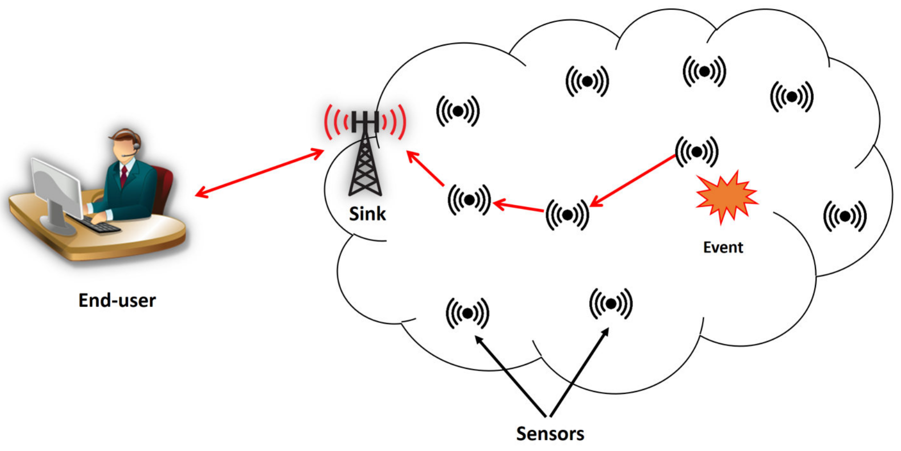


Figure I.1: Wireless Sensor Network

As can be seen from Figure I.1, a WSN comprises a large number of sensors that are placed in a given geographical area. The sensors are required to cooperate to accomplish a common task. The latter can be the surveillance of a battlefield in military applications, the surveillance of a harsh environment, the surveillance of a car park, the surveillance of a forest, etc. Each sensor deployed in a surveillance area must constantly monitor the surrounding environment to detect the events and transmit the information about them to the sink using multi-hop communication.

I.2.1 Architecture of Wireless Sensor Node

The wireless sensor node relies on a set of components in order to perform its duties perfectly. A sensor node consists of a sensing unit, processing unit, communication unit, and power unit. Their different functions are outlined below [11]:

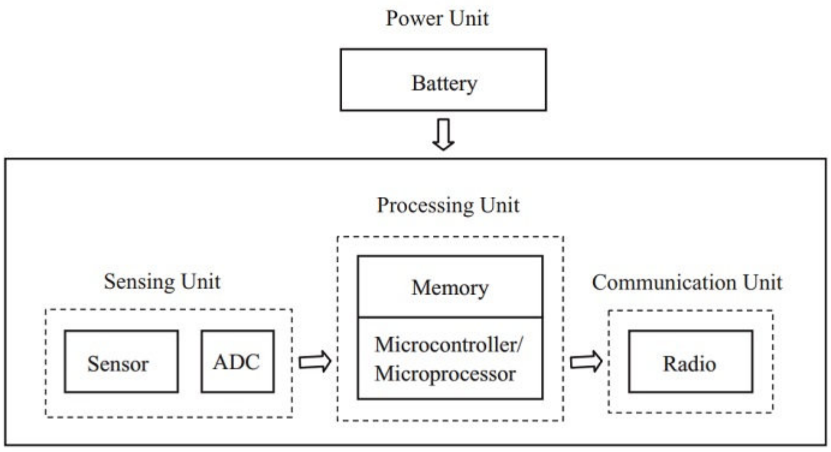


Figure I.2: The components of a sensor node

I.2.1.1 The sensing unit

It is composed of two components, the first one is the sensor, which is generally embedded inside the node, and its primary mission is to collect the data about the events that take place in the node's surroundings. The second component is an analogue to digital converter (ADC), which is responsible for converting the collected information by the sensor to a set of signals that the processing unit can understand.

I.2.1.2 The processing unit

It usually consists of a limited storage unit used to store the data and a processing unit associated with a microcontroller for data processing. This unit is considered the main component of a sensor node because it links the sensing unit and the communication unit. In the processing unit, the underlying operating system is responsible for delivering and receiving instructions from the sensing and communication units through micro-device drivers. Besides, this unit manages the activities that make the sensor collaborate with the other sensors to perform the assigned sensing tasks.

I.2.1.3 The communication unit

It is responsible for connecting the sensor node to the wireless network. This unit manages all the communications, where it receives the incoming signals from the other sensors and passes them to the processing unit, and at the same time, it sends the outgoing signals via radio waves to the neighboring sensors. Several studies showed that the communication unit consumes a big portion of sensors' energy. Also, it is observed that the energy consumption will increase more with the increase in the transmission distance. This is why WSNs use sensors with limited transmission range to save energy.

I.2.1.4 The power unit

It is responsible for energy supply to all the aforementioned units. Without this unit, all the other units will not operate, and all the functions of the sensor will fail. In general, wireless sensors are equipped with a battery as an energy source. Once a sensor is deployed, the battery will not be charged again or changed. However, in some cases, other power sources are used to recharge the batteries, such as solar energy. The different components of a sensor and the relations between them are illustrated in Figure I.2.

I.2.2 Types of nodes in WSNs

To perform the monitoring task, WSNs employ several devices, each of which is assigned a specific job to accomplish such as, sensing data, relaying data, aggregating data, and transmitting data for long distances. The different types of nodes are static sensor node, mobile sensor node, a sink node, relay node, and cluster head node. Their characteristics are detailed below.

I.2.2.1 Static sensor node

The static sensor refers to the node that does not change its initial location during the network lifetime. Static sensors generally operate in indoor and accessible environments, where they are placed in exact locations by humans or using robots. The main drawback of using static sensors is that they are not adaptive to the sudden changes that may occur in the network topology, such as the failure of a sensor [12].

I.2.2.2 Mobile sensor node

The mobile sensor, as its name implies, has all the characteristics of a static sensor. Besides, it has a valuable feature, which is the ability to change its location at any time. Mobile sensors are useful in many situations. For instance, when the failure of a node causes the loss of network connectivity, the mobile node can be moved to the failure location to accomplish the recovery task [12].

I.2.2.3 Sink node

The sink node also referred to as the base station, is a special node with adequate resources. The sink node is responsible for data collection, and sometimes it acts like a gateway to the outside world [13]. There are two types of sink nodes: static sink and mobile sink. The static sink is typically placed in a predetermined location in the sensing area. The sensors in the network use multi-hop communication to forward the collected data to the static sink. On the other hand, the mobile sink changes its position frequently, where it regularly visits all the sensors in the network to collect the gathered data [14].

I.2.2.4 Relay node

Relay nodes are another special type of nodes characterized by extra storage space and powerful transceivers. Their main job is to forward data for long distances in large geographical areas. By using relay nodes, the energy of a regular sensor is saved by concentrating it only on sensing and gathering data rather than depleting it to forward the data gathered by the other sensors [15].

I.2.2.5 Cluster head node

In some applications, the sensors are divided into multiple segments called clusters. In each cluster, a special node is designated to collect and aggregate the data from the nodes within the cluster, and then report the aggregated data to the sink. This special node is known as the cluster head. The cluster head acts as a relay between the sensors and the sink. By making only the cluster head communicate with the sink, the communication overhead is reduced because the cluster head will handle all the communication between the sensors in the cluster and the sink [16].

I.3 Wireless Sensor Network Architecture

Based on the WSN's task and the environment, the wireless nodes are organized in a logical structure known as the network architecture. The architecture defines a set of rules for the sensing devices to follow. The rules help the devices to coordinate in order to gather and transmit valuable data to the end-user. The WSN architectures are classified into four categories as follows: flat architecture, clustered architecture, chain-based architecture, and tree-based architecture.

I.3.1 Flat Architecture

Flat architecture also referred to as unstructured architecture is considered the first architecture used in the early wireless sensor networks. The main characteristic of flat architecture is the absence of any defined structure, which makes the management of the WSN very easy. In this architecture, the network is constructed by placing a number of homogeneous sensor nodes that have the same capabilities and functionalities. Besides, all the sensors perform the same sensing tasks except for the sink node, which is considered a special node with additional resources [17]. Sensors can directly communicate with the sink to report the gathered data or they can use multi-hop communication through passing the data from one node to the other until it reaches the sink. The flat architecture is depicted in Figure I.3.
The use of flat architecture brings several advantages, such as a minimal overhead to maintain the WSN's infrastructure. Moreover, due to the unstructured nature of the topology, each sensor broadcasts the sensed data to all the sensors that lie in its communication range. This gives the sensor multiple routes to forward the data, which is beneficial for fault tolerance [18]. The main drawback of flat architecture is the fast depletion of energy because the nodes need to broadcast data continuously. Another drawback that must be highlighted is that sometimes two

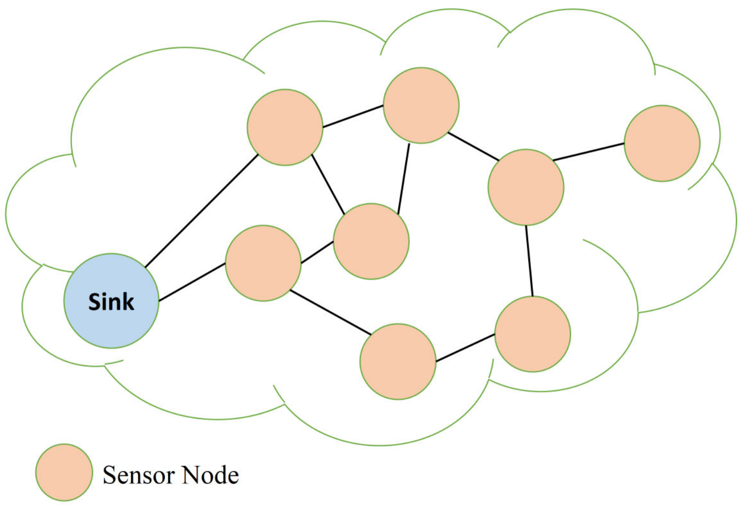


Figure I.3: Flat Architecture

or more sensors will receive duplicate information from others covering the same area, which results in reporting repeated data to the sink.

I.3.2 Clustered Architecture

In contrast to flat architecture that is characterized by the absence of any defined structure, clustered architecture is a special type of network topology that imposes a structure on the WSN. In clustered architecture, the network is formed by a number of heterogeneous sensor nodes with varying capabilities and functionalities. These sensors are divided into several groups called clusters, where each cluster consists of several normal nodes, in addition to a more expensive node with more powerful resources as their leader. The leader node is called the Cluster Head (CH), its main task is to collect and aggregate the data from the nodes within the cluster and then report the aggregated data to the sink directly or via other cluster heads. In a cluster-based structure, the interconnection between the cluster heads forms the backbone of the WSN through which the data is forwarded to the sink node [19]. The cluster-based architecture is shown in Figure I.4.

A clustered network architecture has several advantages, including low communication latency, data fusion, secure data communication, collision avoidance, and energy efficiency. Furthermore, the clustered architecture is used as a solution to scale down the large sensor networks that are organized in flat architecture to make the network operations more efficient [19].

The weakness of the clustered architecture is the fast depletion of the energy of cluster heads because all the communications with the sink are handled by them. This will increase the consumption rate of their energy, making it depleted sooner than the energy of the other nodes [20].

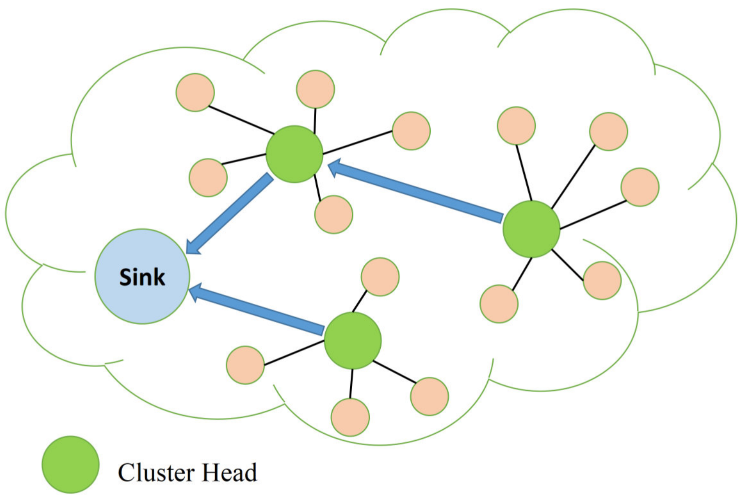


Figure I.4: Clustered Architecture

I.3.3 Chain-based Architecture

In chain-based architecture, the sensors are connected to each other forming a chain structure. Each sensor in the chain transmits data only to its direct neighbors. In each chain, a sensor node is selected to be the chain leader. It is responsible to aggregate the data received from the sensors in the chain and then transmits the aggregated data to the sink node. When a sensor in the chain wants to transmit data, it sends it to the nearest neighbor, then, the data is forwarded from the neighbor to another sensor, which in turn sends it to its neighbor and so on until it reaches the chain leader. This architecture is suitable for applications that monitor bridges, utility pipelines, or railways [21]. The chain-based architecture is depicted in Figure I.5.

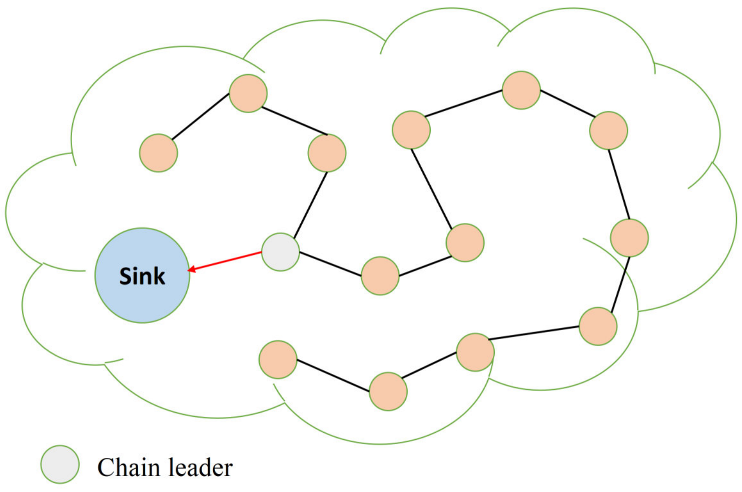


Figure I.5: Chain-based Architecture

Generally, in chain-based architecture, the sensor starts by selecting its nearest neighbor before

it sends data. To do this, a sensor uses the signal strength to measure the distance between him and his neighbors. Based on the calculated distance, the sensor adjusts the signal strength so that only the closest neighbor will receive the signals. This mechanism will save the sensor's energy because each node selects the path with the minimum distance to send data [18].
The critical issue facing the use of chain-based architecture is that once a sensor in the chain fails, the whole chain will stop working [22].

I.3.4 Tree-based Architecture

The key idea behind the tree architecture is to organize the sensors in a layered logical tree structure. Each sensor in the middle layers is called a non-leaf node, and it has a set of children directly connected to it. The sensor nodes located in the lower layer are called leaf nodes. These nodes have parents but do not have any children. The sensor located in the high layer (the root of the tree) is called the leader node. Similar to the chain architecture, the responsibility of the leader is to aggregate the data received from the sensors in the tree and then transmit the aggregated data to the sink node. The flow of data starts from the sender and goes up layer after layer until it reaches the leader. When a non-leaf node receives data from its children, it forwards it to its parent until it reaches the leader [23]. Upon receiving data from one of its children, if the parent node has another data to send, the parent will aggregate the original data with its own data and then forward it to the next layer. The tree architecture is illustrated in Figure I.6.

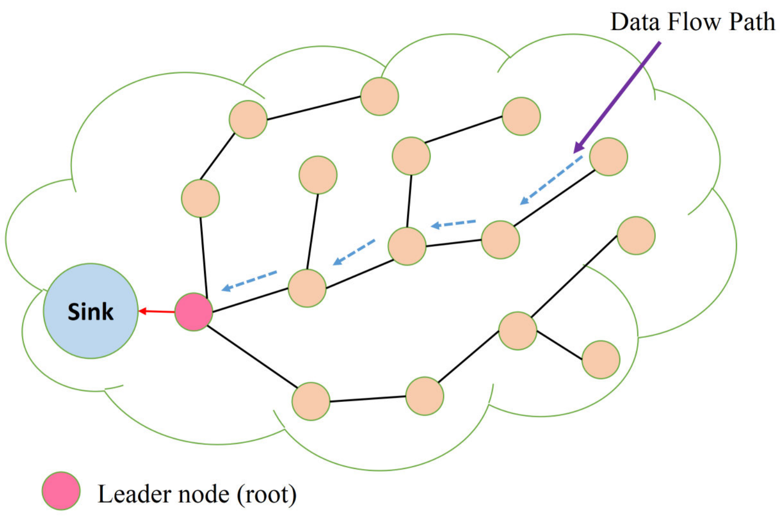


Figure I.6: Tree-based Architecture

The main drawback of the tree architecture is the non-uniformity of energy consumption of sensors in the network. The energy of the non-leaf sensors is depleted faster than the energy of the leaf nodes. The reason behind this imbalance is that the energy of the non-leaf sensors decreases rapidly due to extensive transmission of the data coming from the lower layers [24].

I.4 WSNs characteristics

Wireless sensor networks have a set of unique characteristics that we cannot find in the other types of networks. Their main characteristics are as follows [25, 26]:

- **Cooperative operation :** the nodes engaged in forming the wireless sensor network collaborate to forward packets to their ultimate destination, where every node in a communication path between two nodes behaves like a relay.
- **Dynamic network topology :** the physical topology of the wireless sensor network is frequently changing due to the movement of nodes. The connected nodes can freely leave the network, and new nodes may join the network anytime. In addition, the next hop of each node is determined dynamically based on the new network topology. At any change, the routing tables of nodes are altered to consider the new updates.
- **Flexibility :** wireless sensor networks are characterized by the fast and easy integration of new nodes into the network. Moreover, they are capable of serving a suddenly increased number of nodes without any outside intervention or a modification in the installation.
- **Mobility :** unlike wired networks that force the nodes to stay in one location, wireless sensor networks give the nodes freedom in the movement without losing the connectivity as long as they stay in the communication range of at least one connected node.
- **Cost-effective :** wireless sensor networks are cost-effective solutions because they do not use any kind of wired links between the nodes. The absence of wires decreases the network cost to a minimum value. Besides, no maintenance cost is needed.
- **Scalability :** wireless sensor networks vary in scale from several nodes to potentially several hundred or even a thousand in large networks.

I.5 Types of WSNs

In the early days after the emergence of WSNs, the typical use of WSNs was limited to terrestrial applications where the sensors are placed on the land to collect data. This type of network is known as terrestrial WSN. Currently, WSNs are also installed in other environments including, underground and underwater. Depending on the application, we can differentiate between five types of WSNs: terrestrial WSN, underground WSN, underwater WSN, multimedia WSN, and mobile WSN. Figure I.7 shows the different WSN types.

I.5.1 Terrestrial WSNs

Terrestrial WSNs consist of several sensor nodes that typically varies from a few to thousands. These sensors are placed above the ground either deterministically by relying on a predefined plan or randomly where they are generally scattered from planes over the sensing area. Random placement is used to place the sensors in harsh environments in which survival is difficult or impossible. The main issue facing terrestrial WSNs is energy conservation because

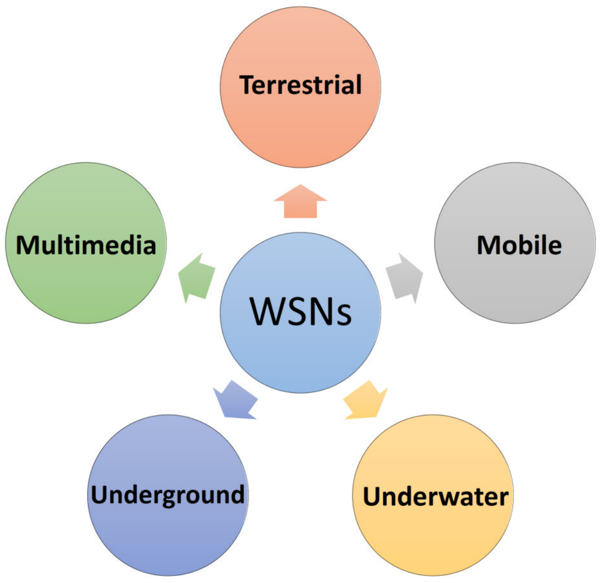


Figure I.7: WSN Types

the wireless sensors use batteries for power supply, which is considered a limited energy source. To alleviate this issue, many applications place terrestrial sensor nodes that are equipped with a secondary power source such as solar cells. Moreover, multiple design solutions are proposed to conserve the energy of terrestrial WSNs including, multi-hop routing, short transmission range, eliminating data redundancy, and minimizing delays. Terrestrial WSNs are used in many fields including, environmental monitoring, industrial monitoring, and surface explorations [27]. An example of terrestrial WSN is shown in Figure I.8.

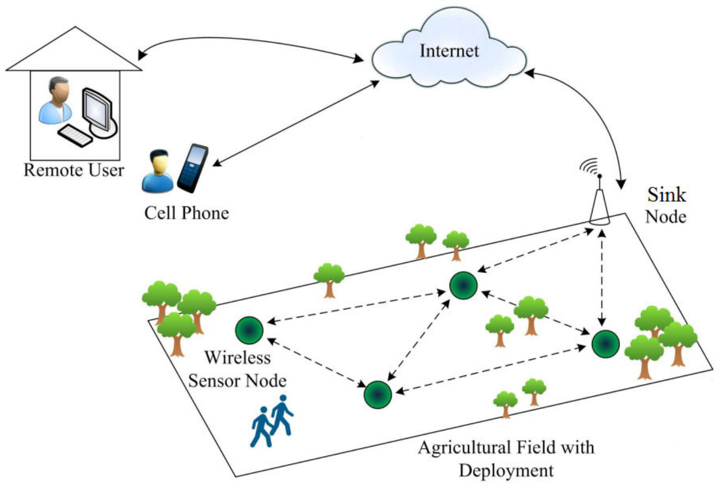


Figure I.8: Terrestrial WSNs

I.5.2 Underground WSNs

Underground WSNs are composed of a collection of sensors that are typically buried in the soil or placed in inaccessible regions underground such as caves. Therefore, the entire network is installed underground, except for one or more wireless sensors that are located above ground to transmit information from the WSN to the sink node. The sensors installed in underground WSNs are known for their high cost compared with those used in terrestrial WSNs. The use of expensive wireless sensors is a basic requirement in underground WSNs because the ground contents such as soil, rocks, water, and other solid contents make wireless communication a difficult task and lead to signal losses due to the high levels of attenuation. The problem of energy is always present even in underground WSNs. This problem is more critical in this type of WSNs because, in contrast to terrestrial WSNs, the sensor battery cannot be charged, and replacing it becomes a very difficult task. There are several applications of underground WSNs, such as agriculture monitoring, underground structural monitoring, underground environment monitoring, and military border monitoring [27, 28]. An example of underground WSN is shown in Figure I.9.

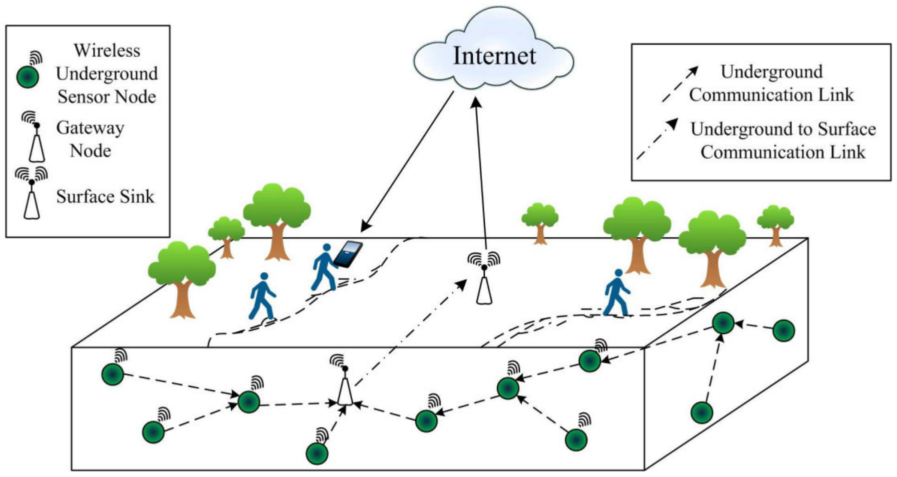

Wireless Underground Sensor Node
Gateway Node
Surface Sink
Internet
Underground Communication Link
Underground to Surface Communication Link

Figure I.9: Underground WSNs

I.5.3 Underwater WSNs

In underwater WSNs, a limited number of expensive sensor nodes is sparsely placed in underwater environments such as seas, lakes, and oceans. This type of network is an interesting research field due to the difficulties encountered while designing the network. The first issue facing the design of underwater WSNs is energy conservation because the sensors' batteries cannot be replaced or recharged. The second issue is related to underwater wireless communication. In general, underwater wireless communications are established through the transmission of acoustic waves. However, it is observed that the acoustic waves are heavily attenuated and altered in water, especially when traveling for long distances. As an alternative, optical communication in green/blue wavelengths is used for transmitting data. Compared to acoustic waves, optical communication offers high band communication and faster propagation in water but only for small distances. Underwater WSNs are used for pollution monitoring, undersea surveillance, and disaster prevention [29]. An example of underwater WSN is depicted in Figure I.10.

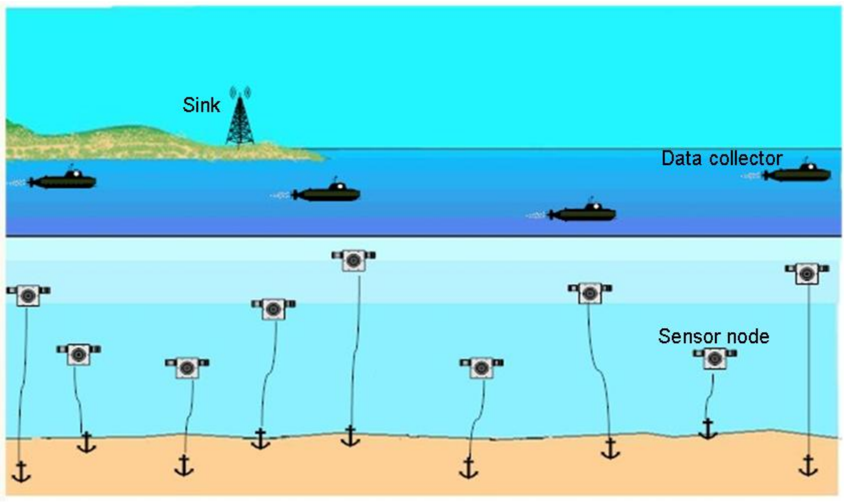


Figure I.10: Underwater WSNs

I.5.4 Multimedia WSNs

Multimedia WSNs are an assemblage of sensors that communicate wirelessly to report data in the form of multimedia such as video, audio, and images. Multimedia WSN usually consists of tiny and low-cost sensors equipped with cameras and microphones. These sensors are deterministically placed at accurate locations that enable them to monitor the events or track targets. Having a higher bandwidth is a requirement in multimedia WSNs because the transmitted content including, images and video stream requires sufficient bandwidth to be delivered. Furthermore, the delivered multimedia content should have a high quality in order to extract useful data from it. Therefore, good quality of service is another requirement in multimedia WSNs. However, the fulfillment of these requirements imposes additional costs where the energy consumption becomes very high in this type of WSNs. Multimedia WSNs are very useful for target tracking, habitation monitoring, traffic management systems, and ecological monitoring [30]. An example of multimedia WSN is shown in Figure I.11.

I.5.5 Mobile WSNs

The mobile WSNs are the most popular networks among all the previously mentioned WSNs. Mobile WSN is a collection of mobile sensors that can autonomously move and reorganize the network. Mobility is the main characteristic of mobile sensors that enable them to change their positions anytime during the network lifetime. Generally, the mobile sensors are initially placed at random, and then they spread out and organize themselves into a cooperative network to perform the monitoring task. The mobility feature enables the sensors to respond quickly to any sudden topology changes which may occur due to the failure of one or more sensors. Mobile WSN is typically used when monitoring an inaccessible area or in harsh environments. The

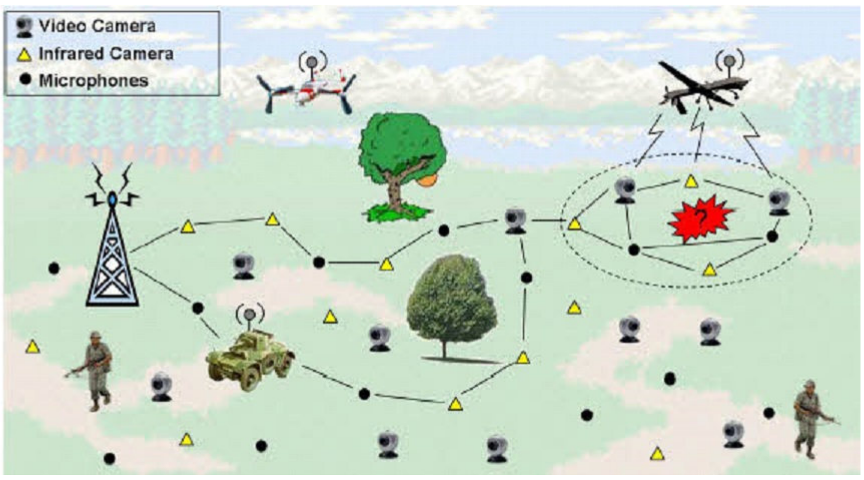


Figure I.11: Multimedia WSNs

applications of such networks are numerous, including the following: meteorology, environment monitoring, infrastructure protection, target tracking, and tactical military surveillance [31, 32]. An example of mobile WSN is illustrated in Figure I.12.

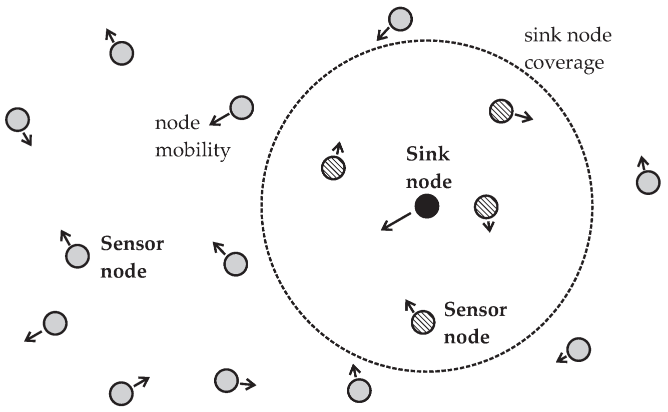


Figure I.12: Mobile WSNs

I.6 Wireless sensor networks protocol stack

The constrained nature of sensors must be taken into consideration when designing the communication protocols for wireless sensor networks. The protocol stack that was developed for conventional computer communication networks cannot be used directly for WSNs due to several reasons including, the dynamic nature of WSNs, the absence of IP addresses in WSNs, and the limited energy sources of WSNs. Therefore, the design of a reliable protocol stack is important for supporting the various requirements of WSN applications. The protocol stack in WSNs consists of the application layer, the transport layer, the network layer, the data link layer, and the physical layer. The functionalities of each layer in the WSN protocol stack shown in Figure I.13 are reviewed below.

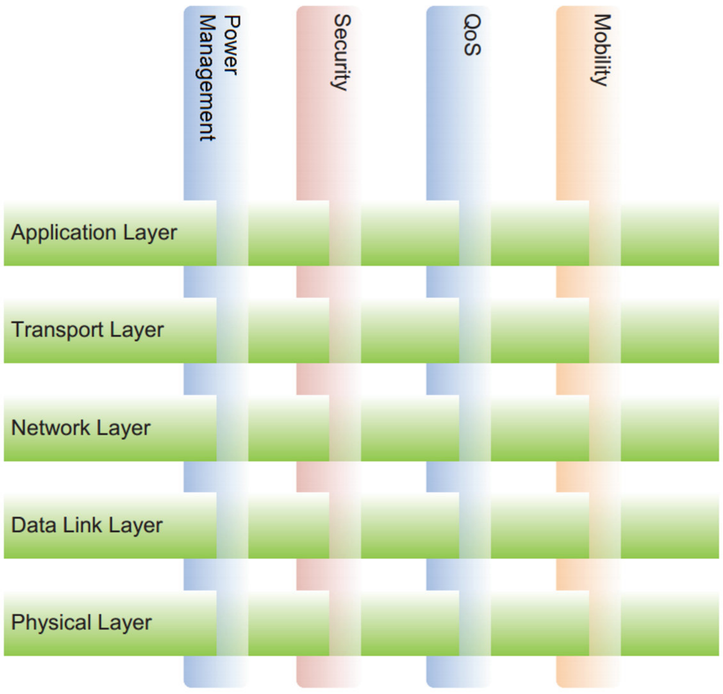


Figure I.13: Protocol stack of WSNs

I.6.1 Application Layer

The application layer supports the different application services and offers several management functionalities for the WSN system administrators. These functionalities include synchronizing the time of sensors, moving sensors, turning sensors on and off, and viewing the status of sensors. Besides, several techniques are performed in the application layer, such as data aggregation, data fusion, and appending a timestamp to the data [33].

I.6.2 Transport layer

The transport layer is composed of several protocols that run over the network layer in order to enable end-to-end message transmission from a given source node to a destination node. Each protocol that runs in the transport layer has to ensure data reliability, packet loss recovery, congestion control, and delivering the packet in the same order. Moreover, the protocol should be designed to support multiple applications. This can be achieved by developing generic protocols that are independent of any specific application. While designing the protocol, the following concerns must be addressed [27]:

- **Data reliability and loss recovery:** the protocol has to ensure the successful transmission of all the packets. Packet loss is inevitable in wireless communications due to several reasons including, poor radio communication, packet collision, node failures, and congestion. Minimizing packet loss is considered a requirement because frequent packet loss degrades the quality of service (QoS) and result in fast depletion of energy. Therefore, immediate loss detection will speed up packet recovery, which in turn prevents the waste of energy.

- **Same order delivery:** due to the node mobility and congestion, packets arrival encounters varied time delays, which results in the packets being delivered out of order. The packets have to be delivered in the same order to guarantee data integrity.

- **Congestion control:** congestion control is a crucial issue in WSNs that requires special attention due to its influence on the QoS and energy consumption. Congestion is usually caused by two reasons. The first one happens at the node level, in which the packet arrival rate exceeds the memory capacity and results in a buffer overflow, which forces the node to drop the packet that will arrive next. This problem is usually encountered at the nodes that are located close to the sink because they handle almost all-upcoming traffic to the sink. The second one is related to the wireless channel conditions, and it is caused by multiple factors such as contention and interference. The protocols that operate at the transport layer must have a set of mechanisms to avoid or control congestion.

The literature provides multiple protocols that operate in the transport layer including, Sensor Transmission Control Protocol (STCP), Price-oriented Reliable Transport Protocol (PORT), and GARUDA.

I.6.3 Network layer

The main task of the network layer is data routing across the network. This layer handles the routing of the data provided by the transport layer. The network layer decides which sensor node to talk to next to forward the data to the sink. The network layer relies on a set of routing protocols to accomplish its task. In contrast to the traditional routing protocols, the protocols used in WSNs are not IP-based protocols because sensors do not have Internet Protocol (IP) addresses. The routing protocols in WSNs have to take into consideration the constrained nature of WSNs. The limited energy, the limited communication bandwidth, the limited memory, and the limited computation capability of sensors must be taken into account while designing routing protocols [27].
Due to the absence of IP addresses in WSNs, one commonly used strategy to route the data is flooding. The principle of flooding is simple, upon receiving data, if the max hop lifetime of the data has not been reached and the receiving node is not the destination; the data is transmitted by broadcasting it to all the neighbors. Although the flooding provides a simple way to transmit data with no costly complex route discovery, it suffers from critical shortcomings including [33]:

- **The transmission of repeated data:** the destination node will receive many copies of the same data, especially when it has a large number of neighbors.

- **Energy wastage:** flooding results in fast depletion of energy due to the useless repeated transmission of data.

To overcome these shortcomings, another strategy called gossiping is used to route data across the network. In gossiping, upon receiving data, a sensor node chooses a random node from its neighbors and forwards the data to it. Gossiping routing algorithm avoids the transmission of

repeated data, however, it suffers from a considerable delay while routing data to the destination.
The existing routing protocols in WSNs are more sophisticated than flooding and gossiping. These protocols consider the constrained nature of WSNs and can handle the routing of data using limited resources. The network layer protocols include but are not limited to Anchor Location Service (ALS) protocol, Secure Routing (SecRout) protocol, and Secure Cell Relay (SCR) protocol.

I.6.4 Data link layer

The principal mission of the data link layer is data transfer between two sensor nodes that share the same wireless link. Actually, this mission is handled by a sub-layer called the Medium Access Control (MAC), which relies on a set of protocols to ensure the successful transmission of data. The protocols that operate in the MAC layer have to handle the establishment of communication between the source and destination. The destination is sometimes thousands of nodes rather than one. Therefore, the protocols have to adapt to the scalable nature of WSNs. Furthermore, since nodes share the same link, a fair share of communication resources between the nodes is needed. The MAC protocol has to coordinate the link access among competing sensors in order to avoid collisions and minimize the number of retransmissions. This contributes to saving the energy of sensors and reducing data loss. In addition, the MAC layer performs more tasks including, frame synchronization, bandwidth utilization control, flow control, and error control [34].
In general, the protocols in the link layer should enhance the quality through maximizing network throughput, enhancing transmission reliability, and most importantly conserving the energy of sensors. Some of the existing data link layer protocols include TRaffic-Adaptive Medium Access protocol (TRAMA), Berkeley Media Access Control (B-MAC) protocol, Low power reservation-based MAC protocol, and Low power distributed MAC protocol.

I.6.5 Physical layer

The physical layer consists of a set of transmission technologies that are responsible for sending bit streams over the wireless link. In addition, in cooperation with the MAC layer, it performs frequency selection, modulation, channel encoding, and error detection and correction. All these tasks should be performed in an energy-efficient manner to prolong the network lifetime. The energy is mainly used in this layer to supply the radio circuitry with power and for transmitting data bit streams over the wireless channel. Regarding radio circuitry, their energy consumption is fixed. On the other hand, the energy consumed to transmit the bit streams can vary based on several factors including, channel loss and interference. The channel loss is affected by many transmission parameters such as modulation method, transmission power, and transmission distance. Among the three parameters, the transmission power and modulation method are the important parameters that must be chosen properly. Transmission distance also has an effect on energy consumption, however, WSN generally uses multi-hop routing, which allows the sensors to conserve energy through transmitting data for short distances from one node to another until it reaches the sink. Regarding the two other parameters, firstly, there is a tradeoff between transmission power and error. When the transmission power increases, the probability of error decreases, which in return contributes to decreasing the amount of energy consumed during retransmission of the loosed data. Secondly, the modulation method is another parameter that a network designer can adjust. Due to the use of wireless transmission channels, modulation methods are needed to transmit bit streams over a wireless channel. The choice of a proper modulation method can increase the success probability when transmitting data and minimizes the energy consumption during the transmission. The existing modulation methods include binary modulation, Multi-Frequency Shift Keying modulation (M-FSK), and Multi-Phase Shift Keying modulation (M-PSK) [27, 35].

I.6.6 Cross-Layer design for WSNs

As explained above, each layer in the protocol stack performs its responsibilities independently, with little or no communication with the other layers. The cross-layer design allows communication between protocols belonging to different layers, which gives them the ability to exchange information and cooperate to achieve a common optimization objective. Therefore, in cross-layer design, the parameters, status, and other information are shared between the layers without breaking the five-layer structure of the protocol stack. The objectives of the cross-layer design include reduction of energy consumption, the guarantee of QoS constraints, mobility, security, and efficient routing. Taking QoS as an example, the application layer can define the QoS criteria, and all lower layers have to follow and ensure the fulfillment of the defined QoS. Moreover, the network layer can utilize the information supplied by the transport layer regarding congestion and channel status information from the physical layer to choose the best route for optimizing energy consumption. Therefore, the cross-layer design improves the overall performance of the WSN by allowing coordination between the different layers [36, 37].

I.7 Wireless Sensor Networks Standards

In the last few decades, several communication wireless standards have been developed, taking into account the WSN's constraints, such as energy and cost. These standards consist of a set of functions and protocols that work together to provide reliable communication environments for WSNs. Some of these standards are IEEE 802.15.4, ZigBee, WirelessHART, and Wibree. In the following paragraphs, we will give a detailed description of these standards.

I.7.1 IEEE 802.15.4

IEEE 802.15.4 is a standard developed under the standardized low rate wireless personnel area network (LRWPANs) concept that aims to standardize the communication protocols and eliminate the need for proprietary technologies that are often designed for specific applications. IEEE 802.15.4 standard is characterized by maintaining a low level of complexity, low deployment cost, and extremely low power consumption during wireless communications. IEEE 802.15.4 is dedicated to applications that only require short-range communications between 10 and 20 meters. The use of a short communication range helps to conserve the battery lifetime, which allow the sensors to remain functional for a longer time. IEEE 802.15.4 is designed to operate in both low and high-frequency bands. The low-frequency bands are 868 MHz that was specified to operate in Europe and the 915 MHz band that operates in North America. Unlike low bands, the high-frequency band spans from 2.4 to 2.483 GHz, and it is used worldwide. The sensing devices that implement IEEE 802.15.4 standard are formed either in a star topology or peer-to-peer topology. In the star topology, all the communications between the sensing devices are passed by a central device, which acts as a coordinator that manages all the star communications. The peer-to-peer topology allows the formation of ad-hoc and self-organizing WSNs, in which a sensing device can communicate with all other devices that are located within its communication range. The MAC sub-layer in IEEE 802.15.4 controls the access to the physical channel and provides a set of services include frame delivery, frame validation, and network synchronization [38].

I.7.2 ZigBee

ZigBee is a global wireless communication standard that provides to a variety of applications a set of features such as cost-efficient network design, effective short communications, low power consumption, and simple implementation. ZigBee is considered an extension of the IEEE 802.15.4 standard because it is built on the defined standards by IEEE 802.15.4 for both the physical and the MAC layers. ZigBee defines the standards and functionalities for the upper layers that are not considered by the IEEE 802.15.4. ZigBee gives more scalability to the network compared with IEEE 802.15.4 because ZigBee can have up to 653356 connected devices. Furthermore, the communication range of each device can reach 50 meters. Therefore, the ZigBee technology allows the formation of very big WSNs. Generally, the ZigBee devices utilize multi-hop communication in order to forward the data across the big network. Besides, ZigBee uses direct sequence spread spectrum (DSSS) modulation to provide reliable data transmission services. The ZigBee networks are formed in mesh, star, or cluster tree structures. The ZigBee tree topology is composed of three types of devices: the coordinator, the router, and ordinary devices. The coordinator is a special device used as a gateway to the outside, it manages network nodes, and sometimes it acts as a bridge to other networks. The routers are intermediate devices that are responsible for relaying data in the network. They are also used to extend the network over a large geographical area. The ordinary devices are sensors equipped with low-power batteries and designed with high energy-saving capabilities. Zigbee caught the attention of a wide range of applications including, commercial applications, industrial applications, and even governmental organizations [39, 40].

I.7.3 WirelessHART

WirelessHART is a communication standard mainly designed for process measurement in industrial applications. Although the other wireless standards such as ZigBee are considered a feasible solution for many applications, they cannot meet the stringent requirements of industrial applications. WirelessHART is designed to be easy to deploy, reliable, secure, scalable, and energy-efficient. The WirelessHART physical layer is mostly based on the IEEE 802.15.4 standard specifications. WirelessHART operates at a 2.4GHz band with a data rate of up to 250 kbits/s. Also, it utilizes TDMA technology to avoid collisions and employs the channel-hopping strategy for choosing the channels that are free of interferences. WirelessHART networks are generally formed in a mesh topology to provide multiple paths for routing data in environments that contain physical obstacles and high interference. Security is an important feature provided by WirelessHART. The transport and network layers provide end-to-end secure communication using encryption, authentication, and key management. In the WirelessHART network, several types of devices are deployed each of which is responsible for performing a specific task. The first type is called field devices, they are wireless devices attached to the plant or process equipment. Handhelds are portable WirelessHART devices used for the installation, configuration, diagnostics, and maintenance of all kinds of WirelessHART devices. Gateways are similar to the access points, they are used to connect host applications with WirelessHART network devices. Finally, the network manager is a centralized device responsible for scheduling resources, managing communications, managing routing tables, and configuring/reconfiguring the network [41, 42].

I.7.4 Wibree

Webree is a communication technology that resembles Bluetooth technology because it was originally adapted from the Bluetooth specification. Webree technology operates in the 2.4 GHz band spectrum with a data rate of up to 1 Mbps. Wibree is designed to operate with a transmission range not exceeding 10m. Wibree can work alone or together with Bluetooth, therefore, there is no need to install an extra antenna in the devices because one antenna is sufficient for both technologies. This feature contributes to minimizing the network cost. In addition, compared to all other technologies, Webree offers ultra-low power consumption in both active and idle modes. This helps in maintaining the lifetime of batteries up to 2 years. Wibree offers a reliable point-to-point and multi-point data transfer with advanced encryption functionalities for several devices including, watches, wireless keyboards, mobile phones, multimedia computers, and PCs. Webree technology is used in many applications include sports, healthcare, and mobile and PC accessories [43].

I.8 Operating systems for WSNs

The operating systems of WSNs are designed to operate in the constrained environment of sensors, which is limited in terms of memory, computational power, and energy. The literature provides several operating systems, each of which has a unique design structure and multiple capabilities that enable it to support most of the WSNs applications. In the following paragraphs, we will give an overview of the most popular WSN operating systems, and we will highlight their major design characteristics.

I.8.1 TinyOS

TinyOS is an operating system designed by taking into consideration the limitations and constraints of sensors. TinyOS is an open source and flexible system with a tiny size not exceeding 400 bytes. TinyOS is composed of a set of components that are assembled to form an application-specific system for WSNs. TinyOS is a flexible system that falls under the monolithic architecture class. In this class, a set of components are assembled based on the application requirements. Each component in TinyOS is an independent computational entity that is designed to perform a specific task. TinyOS is implemented using the NesC language. TinyOS supports the lightweight concurrency model and multi-threading, in which concurrent programs are executed using very low memory. Components in TinyOS are expressed in three computational abstractions namely: commands, events, and tasks. The commands and events are mechanisms mainly used for inter-component communication. For instance, a command can be used to request a service from a component, and the event can be used as a signal that indicates the completion of that service. TinyOS make use of tasks to express intra-component concurrency. The operating system provides a multi-hop communication protocol called TYMO, which is designed based on a routing protocol used in mobile ad hoc called DYMO. In addition, TinyOS provides multiple services and features including, a specific file system, distributed services, sensor drivers, database services, security services, and the support of multiple sensing platforms [44].

I.8.2 Contiki

Contiki is a lightweight operating system that offers a rich execution environment for devices with limited resources. Contiki is a highly portable operating system codified using the C programming language. In contrast to TinyOS, Contiki is built following modular architecture. Contiki is composed of the kernel, a set of libraries, the program loader, and a set of application programs and services. In Contiki, all communications go through the kernel, which acts as an intermediate layer that allows drivers and applications to communicate directly with the hardware components. In addition, the kernel operates using the event-driven model, where an event scheduler is employed to dispatch events to running applications. Contiki supports multi-threading but in a different way called preemptive multi-threading. The Contiki multi-threading is implemented as a library that can be linked with application programs that require multi-threading. Contiki uses a simple method for scheduling where the incoming events are fired to the target application as they arrive. Contiki comes with a set of communication protocols that allows applications to communicate easily. Besides, it provides a layered protocol stack called Rime that supports single-hop unicast, single-hop broadcast, and multi-hop communication. Contiki also supports dynamic memory management, security services, simulation services, and multiple sensing platforms [45].

I.8.3 MANTIS OS

MANTIS OS is another lightweight open source operating system designated for WSNs. MANTIS is an abbreviation for MultimodAl system for NeTworks of In-situ wireless Sensors.

MANTIS OS also called MOS, is an easy-to-use and energy-efficient operating system written in the C programming language. The main components of MOS are the kernel, the scheduler, system APIs, and the network stack. The size of all these components does not exceed 500 bytes. MOS is designed using the layered architectural design where each layer represents a service provided by the MOS. MOS provides a convenient environment for creating, testing, and debugging WSN applications. Besides, MOS is a flexible and expandable OS that supports multiple features including, dynamic reprogramming of sensors, remote debugging of sensors, and multimodal prototyping. MOS is a multi-threading OS designed to use a priority-based scheduling mechanism similar to that found in classical UNIX systems. In addition, binary mutexes and counting semaphores are implemented in MOS to avoid race conditions and allow safe resource sharing. In MOS, the layers of the network protocol stack are separated into two parts. The first part consists of the network, transport, and application layers. These layers are implemented in user space to provide more flexibility. The second part contains the implementation of the MAC and physical layers, which have been merged into a single layer called the COMM layer. The main task of the COMM layer is to provide a unified interface for communication with device drivers. MOS provides a set of additional features including, application development in the C language, WSN simulation, and implementation of a Unix-like shell [46].
For further information about other WSN operating systems including, Nano-RK OS and LiteOS, please refer to the following reference [45].

I.9 WSNs applications

WSNs provide numerous features that made it the focus of attention of several applications. Due to these features, WSNs became a modern technology currently installed to perform multiple tasks in several domains. In the following paragraphs, some of the popular WSNs applications including, military, environment, health, and industry, are reviewed in detail.

I.9.1 Military applications

Military operations are the main reason behind the emergence and the development of wireless sensor networks. WSNs are considered an excellent surveillance tool for military applications that assists the military in performing several operations including, intrusion detection, battlefield surveillance, and force protection. A popular use of WSNs is to detect intrusion in a secret military area. Sensors are placed in strategic locations inside the area or in the area borders to sense the environment and alarm the military in case of an intrusion. Another use of WSNs is to monitor the battlefield during the war. Critical regions and points can be closely monitored using sensors to help the military in decision-making and organizing counterattacks by using the collected information about the enemy activities inside that area. Force protection is another important task in military applications that relies on sensors to protect the lives of soldiers. Soldiers can wear the sensors in order to enable the military base station to track their vital functions and detect states of serious distress or risks of fatality. In real applications, a counter-sniper system is developed to detect and locate the positions of shooters (see Figure

I.14). In this system, acoustic sensors are installed to detect the acoustic shock wave that originates from the sound of gunfire. Each sensor that detects the acoustic wave forwards the data to the sink. Then, the sink uses all the incoming data about the acoustic waves and the trajectory of the bullet to guess the location of the shooter [10, 47].

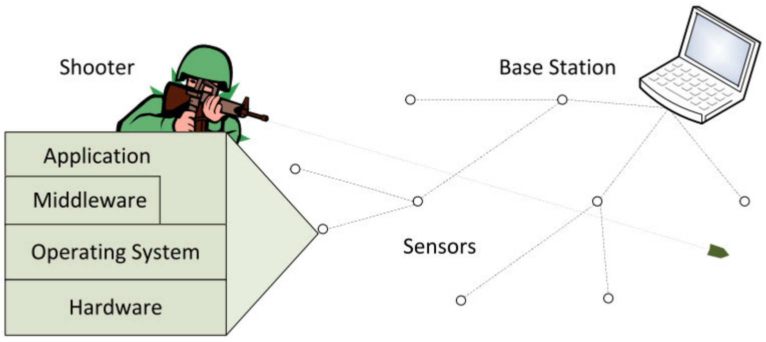


Figure I.14: System architecture in the counter-sniper application.

I.9.2 Environment monitoring applications

WSNs are at the top of the preferred technologies used in several environmental monitoring applications. These monitoring applications typically require real-time measurements of several parameters including, wind, humidity, temperature, and water level, to enable the user to react in order to avoid the numerous losses caused by inaccurate or late measurements. Environmental monitoring applications are numerous in both indoor and outdoor environments. Habitat monitoring is an important task in environment monitoring that enables us to avoid disasters that destabilize the ecosystem as a whole. Ecological disturbances are generally caused by pollution. WSNs are extensively used in habitat monitoring to avoid the negative impacts of pollution by sending alerts in case of any ecological disturbance spotted in animals or plants. Furthermore, WSNs are employed to provide a real-time monitoring for agricultural chambers that are also known as greenhouses. The main objective in greenhouses is to adjust the levels of temperature and humidity to proper levels. WSNs are deployed in greenhouses where the sensors are placed at accurate locations to allow a rapid intervention in case the temperature and humidity levels drop below the threshold value. The sensors send alerts to turn on fans and other systems in order to smoothly adjust the different measurement levels. Another use of WSNs is to prevent natural disasters such as forest fires. The rapid climate change causes the temperature to reach high levels in many places around the world. This abnormal change causes huge forest fires that destroy both animals and plants. WSNs are used as forest fire prevention systems that are capable to provide real-time monitoring to several measurements in order to make quick estimations of fire danger and inform the authorities [48]. Figure I.15 shows a wireless sensor network used for detecting fires in forests.

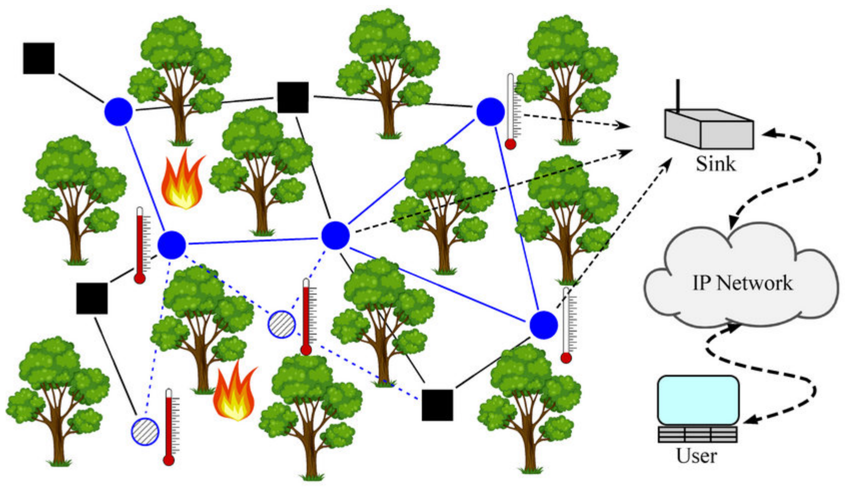


Figure I.15: WSN architecture for forest fire detection.

I.9.3 Healthcare applications

The various features offered by WSNs prompted their use in healthcare systems. Nowadays, a special type of WSNs that consists of tiny sensors having little power is used to monitor health conditions. This type of WSNs is called Wireless Body Sensor Networks (WBSNs). WBSN is composed of wearable tiny sensors embedded in hardware. WBSN monitors the patients' health conditions such as the heartbeat rate, temperature, stress level, and oxygen level and then sends the collected data to the end-user, which is typically a physician (see Figure I.16). Based on the reported data, if the physician spots an abnormal condition, the patient will receive proper treatment before the condition get worsens. WBSN is also used in home assisting systems that are designed for patients. The home assisting systems are utilized to provide personalized medical care assistance for those patients that do not need to stay in the hospital, but still, their health conditions must be monitored. In these systems, the sensors are placed on the body of the patient or in its vicinity in order to record their daily activities. In case of any kind of health crisis, alerting messages are automatically sent to the doctor. For patients that are in critical condition, WSNs are installed in healthcare facilities to provide real-time monitoring for hospitalized patients. WSNs send emergency alerts when the patient's health deteriorates in order to allow doctors and nurses to respond quickly [49].

I.9.4 Industrial Applications

The domain of industrial applications is considered an area of interest where WSNs are applied. Many industrial systems rely on WSNs for monitoring and controlling various parameters in order to acquire better handling capabilities of problems. Industrial systems are numerous, and in all of them, the WSNs can be installed to perform primary or secondary tasks. One essential use of WSNs is in safety systems where immediate action is required in order to enable the system administrators to react faster in case of problems. In these systems, the sensors are placed at predetermined locations that enable them to cover the entire area of concern. Sensors monitor some measurements and send alerts when an abnormal situation is

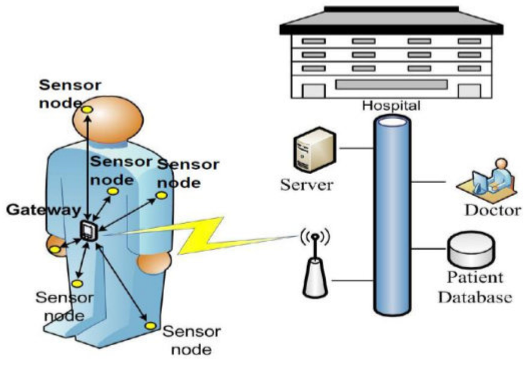


Figure I.16: Architecture of WBSNs in a medical healthcare system.

spotted. An example of such systems is fire alarm systems. WSNs are also utilized to regulate the industrial systems by periodically sending measurements to special equipment called the controller. The reported measurements assist the controller in making decisions and ensure a smooth operation of the system in normal and abnormal situations (see Figure I.17). In addition, WSNs are installed in massive facilities to examine the performance of various types of industrial equipment. This domain is called machinery health surveillance. Sensors are used to detect or predict the occurrence of faults that are obstructive for the equipment work. As a final example of WSN industrial applications, the WSNs features are utilized in the transport logistics domain to ensure the delivery of packages in high-quality by reducing losses during transportation. Sensors are placed in a cargo container with the packages to allow continuous supervision of their status during transportation [49, 50].

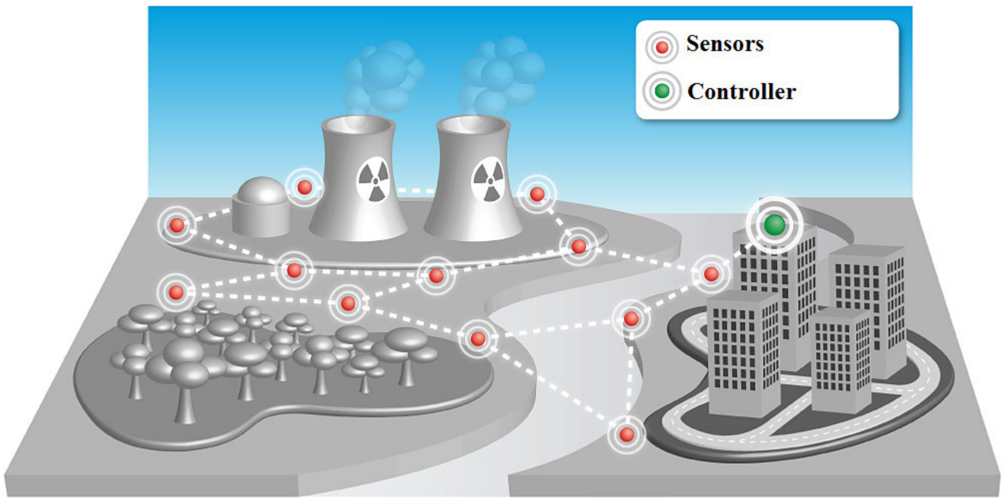


Figure I.17: Architecture of industrial WSN.

I.10 Design issues and challenges in WSNs

WSNs are considered the dominant technology in remote monitoring and data gathering because of the countless advantages they offer for different applications. However, there are still numerous challenges and critical issues facing the network administrators and the designers of applications while setting up a workable and efficient sensor network. In the following subsection, we will give a detailed overview of the major design problems encountered during the setup of a reliable and optimized WSN.

I.10.1 Clustering

In most applications, wireless sensor networks are formed using a large number of sensors that are equipped with limited energy resources. To conserve energy and extend the network lifetime, topology control is a major requirement to be considered during the design of WSNs [51]. Clustering techniques are the most popular approaches used for topology control. Sensor clustering offers many advantages, such as scalability, energy efficiency, and reducing routing delay. Clustering methods achieve these goals by dividing the sensors into several groups called clusters. Within each cluster, a sensor named the cluster head is chosen to be the leader or the coordinator of the cluster. The main challenge in clustering is how to divide the network into groups and how to choose the cluster heads among the sensors. The number of sensors in each cluster has to be chosen carefully to avoid several problems, such as cluster overhead and intra-cluster communication failure. Moreover, optimal cluster head selection can significantly decrease energy consumption, which allows the sensors to remain functional for a longer time. Thus, the distance between the cluster members and their cluster head should be decreased as possible to avoid long-distance communication. Furthermore, proper selection of cluster heads increases the success probability of inter-cluster communications used to forward the gathered data to the sink directly or using multi-hop communications via other cluster heads. Therefore, the design of effective clustering methods is an essential step in large WSNs that contributes to optimizing energy consumption and increasing the lifetime of the networks [8].

I.10.2 Data aggregation

In WSNs, the energy consumed during data transmission is much more than that consumed in performing computation. To avoid fast energy depletion during transmission, data aggregation techniques are applied in WSNs. Data aggregation can be defined as the process of combining the data coming from different sensors and outputting the aggregated data prepared for transmission to the sink node. In large networks, the data transmitted directly to the sink node is massive and cannot be processed efficiently. The transmitted data from neighboring sensors are generally redundant and highly correlated [52]. Therefore, data aggregation techniques attempt to collect the most vital data from the sensors and then forward it to the sink in an energy-efficient manner. Considering data aggregation during the design of WSNs can provide several advantages including, elimination of redundancy, minimizing the number of transmissions, enhancing the Quality of Service (QoS), and reducing energy consumption. There are many challenges while performing data aggregation. Firstly, accuracy and quality of data have to be maintained during data aggregation by eliminating only repeated and useless data. Secondly, the latency and the network overhead should be reduced in order to ensure rapid transmission of data. Thirdly, the data aggregation strategy has to ensure a reliable transmission through employing congestion control mechanisms and reducing the number of hops to reach the sink [53]. All the aforementioned challenges make the development of an efficient data aggregation strategy a fundamental task that facilitates the design of effective WSNs.

I.10.3 Localization

Wireless sensor networks have been used by many promising applications in several domains. In applications that operate in harsh and inaccessible environments, the sensors are randomly scattered over the terrain, which renders the exact locations of sensors unknown. Most applications such as object tracking and data tagging require the information of the place of origin of events to be transferred with the gathered data in order to make decisions and analyze the information efficiently. Moreover, the location information of sensors is also indispensable in many WSN operations including, geographic routing and clustering. Therefore, determining the locations of sensors is a basic requirement in WSNs. Localization is defined as the task of determining the coordinates of sensors in 2D or 3D space. There are many accurate technologies used for sensor localization, such as the Global Positioning System (GPS). However, equipping each sensor with a GPS increases the network cost and energy consumption and degrades the performance of the WSN. As an alternative, localization techniques are used to determine the accurate positions of sensors. The complexity of localization arises from different design factors including, the network architecture, sensor density, geometric shape of the area, and type of sensors (static, mobile). Furthermore, the various application requirements also determine the used localization technique because some applications are satisfied with an estimation of the locations, whereas others need the accurate locations of sensors. Therefore, all these factors have to be taken into account while designing a localization algorithm for WSNs [54].

I.10.4 Fault tolerance

As in any normal system, the faults in WSNs are inevitable. Faults are abnormal conditions of system components that lead to the occurrence of errors if not addressed quickly. Several factors lead to the occurrence of faults in WSNs including, the harsh environmental conditions, the abrupt modification in network topology, lack of power, hardware and software failures, and network bottlenecking [55]. During the design of WSN, careful measures must be taken to allow the WSN to maintain its functionality at an acceptable level even with the presence of faults. Fault tolerance is the ability to sustain the functionality of the WSN even with the presence of interruptions caused by the faults. A proper fault tolerance strategy has to ensure the availability, dependability, and reliability of WSNs operations. In addition to the obvious factors that cause faults, the harsh environment and the limited energy source add another layer of complexity that must be dealt with by putting in place a robust fault tolerance mechanism. The fault tolerance mechanism has to identify and/or predict the occurrence of a failure in the correct time in order to perform a fast fault recovery and allows the WSN to continue its service even after the occurrence of faults [56]. All the aforementioned challenges make fault tolerance an essential step to be considered during the design of WSNs.

I.10.5 Security

The security of wireless sensor networks is of great concern for almost all applications. The constrained nature of WSNs and the limited remote access of the network administrator makes the WSNs very vulnerable and result in threats that expose the sensors to dangerous situations [57]. Unlike wired networks, wireless sensors broadcast their messages to forward the data to the sink. An attacker can easily compromise a sensor, alter the integrity of the data, inject fake messages, and waste its limited resources. Hence, a set of countermeasures must be implemented to address the issue of security in WSNs. The goal of security services in WSNs is to avoid the catastrophic consequences of attacks by protecting the information collected by the sensors. Therefore, designing security protocols that utilize several security mechanisms such as key management, cryptography, certification, and authentication are necessary to avoid the disastrous effects of attacks and help to create a relatively safe working environment for wireless sensors [58]. The security protocols have to ensure all the security requirements, including, confidentiality, integrity, and availability of data because WSNs that are developed without security requirements are easy targets for attackers. The main problem in WSNs is how to implement efficient security services taking into consideration the limited energy, bandwidth, computation, and storage capacity of sensors. This problem remains an open research issue that requires substantial research efforts in order to develop robust security solutions for WSNs.

I.10.6 Deployment

A wireless sensor network is a collection of sensors dispersed in an area of concern to perform the monitoring task. The success of the monitoring task highly depends on the locations of sensors. These locations must be chosen carefully to ensure complete surveillance for the entire area. The planning of network topology and the locations of sensors is referred to as sensor placement or sensor deployment problem. Deployment is a challenging issue in WSN, which directly affects the quality of service of the network. The deployment scheme should be able to determine the number of used sensors and their optimal locations, especially in inaccessible and harsh environments, where the sensors are firstly placed at random, and then the deployment scheme is employed to move the sensors to strategic locations. Besides, the complexity of several problems in wireless sensor networks can be reduced by designing an efficient deployment scheme that optimizes the coverage, connectivity, cost, and network lifetime. Based on the application, the deployment scheme has to determine the optimal placement pattern for sensors in order to satisfy the different requirements [59]. The planning of a sensor placement strategy by taking into account the various application requirements makes the deployment a critical issue in the design of WSNs.

I.11 Conclusion

In the first chapter, we have presented a general overview of the Wireless Sensor Network (WSN) technology, we have reviewed the physical architecture of a sensor node and the different sensor types employed in WSNs. Besides, we have surveyed the architectures, types, protocol stack, standards, and operating systems of WSNs. Moreover, we have highlighted popular applications of WSN in various domains. Finally, we have presented in detail the issues and challenges encountered by administrators and designers while setting up a workable and efficient sensor network that is capable to satisfy all application requirements.
Among the design challenges, the deployment problem has emerged as one of, if not the most critical problem that directly affects the performance of the WSNs. If the deployment of WSN does not consider the intrinsic properties of the quality of service, all the following operations will fail to achieve the application requirements, thereby, the whole WSN will fail to deliver the expected performance.

CHAPTER II

BIO-INSPIRED OPTIMIZATION ALGORITHMS: AN OVERVIEW AND CLASSIFICATION

Contents

II.1 Introduction

Nowadays, many real-world problems have become very complex and hard to solve. The complexity of such problems arises from several reasons, such as the limitation of resources and multi-objectivity. Therefore, the need for specific tools that handle this type of problems is an important requirement. In fact, the literature provides a set of tools called deterministic (exact) methods used to obtain the optimal solution of a given problem. However, these methods are often designed for specific problems and not for all problems. Besides, deterministic methods require a lot of computational time to achieve a satisfactory solution. As an alternative, bio-inspired algorithms can provide appropriate techniques to solve the majority of real-world problems with an acceptable cost through simple mathematical modeling of social intelligence and the concepts of evolution in nature.
In this chapter, we start by defining the optimization process and the combinatorial optimization concept. Then we describe the types of methods used to solve combinatorial optimization problems. We will focus on giving a detailed overview of stochastic optimization, metaheuristics, nature-inspired algorithms, and the two classes of bio-inspired algorithms namely evolutionary algorithms and swarm intelligence. This chapter allows gaining a valuable understanding of the theory of bio-inspired algorithms, which is considered an essential part of our contribution.

II.2 Optimization Problem

In the last few decades, the complexity of real-life problems has increased much more than before, which has given rise to the need for new and efficient optimization techniques [60]. Optimization problems can be found in all fields of applied mathematics, engineering, economics, and other sciences. They are wide-ranging and numerous, hence methods for solving these problems have attracted the attention of researchers.
Optimization can be found in all life aspects, as regular people, engineers, and researchers strive to find the best way to achieve a particular objective. Optimization relies on the concept of optimal thinking in solving problems. Based on this concept, the problems are formalized as optimization problems and a set of optimization techniques attempt to solve them by determining an optimal set of decisions [61]. Optimization is key in solving complex problems in all engineering disciplines. It is the process of finding the optimal solution in the most efficient manner [62]. The ultimate goal of optimization is to determine the best combination of values that minimize the effort required for solving the problem or maximize the desired benefit.
In order to solve most engineering optimization problems, several elements have to be identified and formalized to facilitate the problem solving process. These key elements are outlined below.

- **The decision variables :** During optimization, the optimizer wishes to choose the appropriate decisions for the model being optimized to maximize the benefit or minimize the solution effort. The choices for the optimizer are numerous, therefore, the best choices that lead to the optimal solution in an efficient manner must be chosen. In order to facilitate decision making, a set of decision variables is determined for each optimization problem. In most engineering optimization problems, the decision variables represent some physical dimensions being optimized in the model [63]. All optimization problems require at least one decision variable because without decision variables there is nothing for the optimizer to decide, and thus no problem to solve.

- **The objective function :** Optimization process aims at finding an acceptable or optimal design that satisfies the problem requirements. At each step of optimization, the decision maker will have several decisions to choose from. Each decision will create a unique and different design model. Consequently, there will be more than one acceptable design available, and the task of the decision maker becomes to choose the best one of the many acceptable designs available. In order to make this choice, the decisions must be evaluated to determine the best among them. These decisions are ranging from poor to good based on the design quality. The objective function is used to measure the quality of the decisions by outputting a numerical value that reflects how good the chosen decisions are. Therefore, the objective function is essentially a formulation of a design criterion that the decision maker is trying to achieve [64]. Based on the design goals, the decision maker will either maximize or minimize the objective, e.g., maximize profit, and minimize the delay. Furthermore, through observing the direction of the objective function, we can classify the optimization problem into a maximization problem or a minimization problem.
- **The constraints :** During optimization, the decision maker iteratively attempts to find the best combination of decision variables that minimize or maximize the defined objective function. In many practical problems, the values of the decision variables cannot be chosen arbitrarily because they have to satisfy certain requirements defined by the decision maker. The requirements or the restrictions that must be satisfied are known as constraints. Constraints are usually given by a set of inequalities and/or equalities that impose limits on the decisions that can be made. Although some optimization problems may not have any constraints, most optimization problems involve one or many constraints due to the limitation in the availability of resources or in other functional requirements [65]. Therefore, constraints divide the optimization problems into constrained problems and unconstrained problems. Through constraints, the decision maker can define the set of decision variables that can be feasibly chosen during optimization. This feasible set defines what is known as the feasible region of the problem. The existence of constraints in problems defines a nonempty feasible region that is filled with some restrictions to be followed by the decision maker to produce an acceptable solution for the optimization problem [66].

Based on the three elements described above, the generic form of most optimization problems can be written as:

$$Minimize\, f(x_1, x_2, ..., x_n) \tag{II.1}$$

$$Subject\, to \;\; a_i(x_1, x_2, ..., x_n) = 0 \;\; i = 1, ..., p \tag{II.2}$$

$$c_j(x_1, x_2, ..., x_n) \geq 0 \;\; j = 1, ..., q \tag{II.3}$$

Where $x_1, x_2, \ldots, x_n$ are the decision variables that must be adjusted, the equation (II.1) is the objective function, and equations (II.2) and (II.3) are constraints.

II.2.1 Local Optimum vs Global Optimum

In general, optimization techniques attempt to find the ”best possible” solution in search spaces that frequently have a number of sub-optimal (local) solutions. The existence of such local solutions makes it hard for the optimizers to determine the best (optimal) solution in the search space. In optimization, the best solution is called global optimum, it represents the point where the objective function value is smaller/greater than at all other feasible points in the search space. On the other hand, local optimum is the point where the objective function value is smaller/greater than at nearby points. Thus, the local optimum is the better solution in some feasible neighborhood, but not necessary better than all the points in the search space. Figure II.1 illustrates a local and global optimum of a mathematical function.

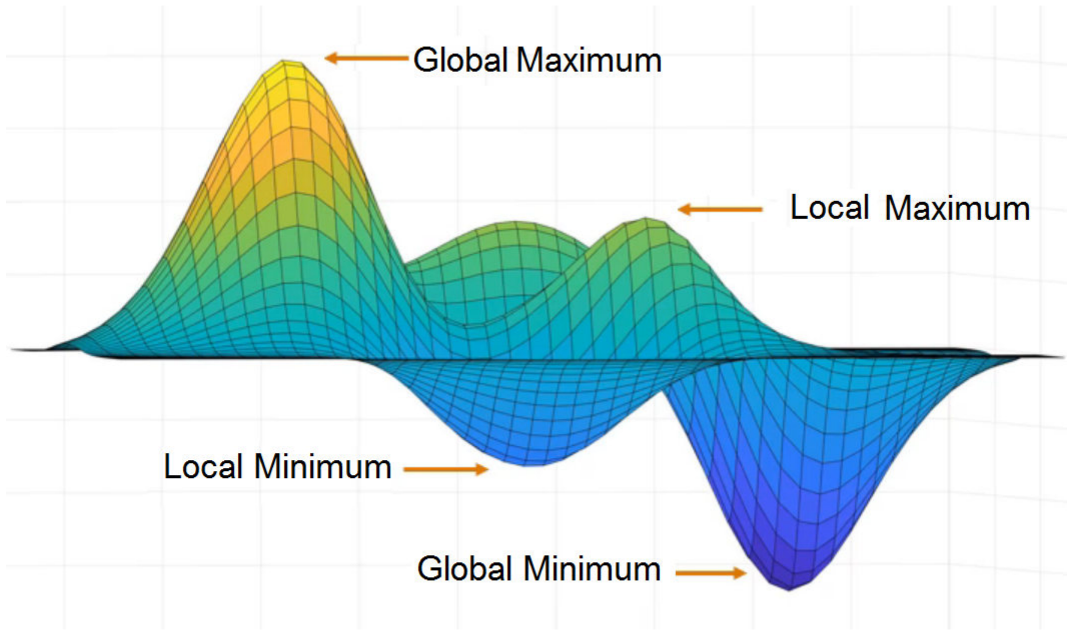


Figure II.1: Illustration of local and global optimum.

II.2.2 Optimization problems classifications

There are several classifications for optimization problems in the literature. Firstly, by changing the nature of decision variables, the problems are classified into continuous versus combinatorial. Secondly, some problems attempt to achieve one objective, whereas others seek several objectives. In this case, the problems are classified into single-objective versus multi-objective. Finally, keeping or eliminating constraints divides the optimization problems into constrained versus unconstrained problems.

II.3 Combinatorial Optimization

Combinatorial optimization is a branch of optimization algorithms related to operations research, algorithm theory, and computational complexity theory, which is devoted for discovering the optimal grouping, order, or arrangement of discrete events with mathematical methods [67]. The goal of combinatorial optimization is to identify an optimal solution from a finite set of feasible solutions. The solutions are normally discrete or can be formed into discrete. The so-called optimal solution is typically an integer number, a subset, a permutation, or a graph structure.

Further, many problems in engineering such as routing, task allocation, scheduling can be modeled in the form of combinatorial optimization problems and solved using different methods. The methods used for solving combinatorial optimization belong to two groups of different nature: exact methods (deterministic) versus stochastic methods (approximate). Figure II.2 illustrates the different optimization methods belonging to each group.

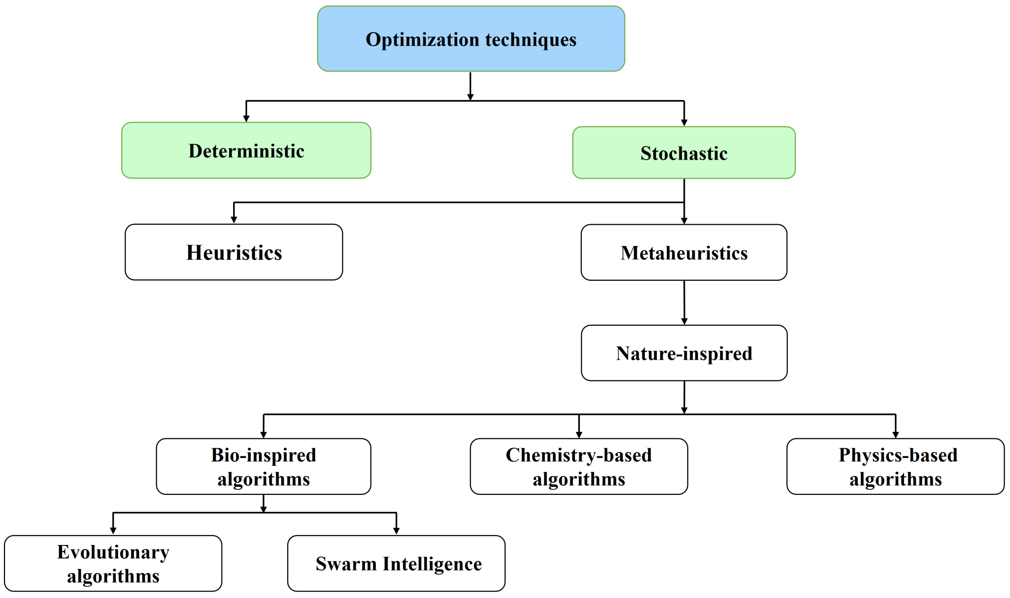


Figure II.2: Taxonomy of resolution methods [85].

II.4 Stochastic optimization algorithms vs deterministic optimization algorithms

Stochastic optimization algorithms are a family of algorithms that is characterized by the use of stochastic operators. Randomness is an important characteristic employed by stochastic algorithms when searching for the global optimum in the problem search space [68]. This characteristic makes stochastic algorithms distinct from conventional deterministic algorithms. In deterministic algorithms, the same answer for a given problem is determined when the optimization starts with the same initial starting point. This will give rise to a serious problem known as the search stagnation problem, in which the algorithm will assume that a local solution is the global solution, and thus it fails to obtain the true global optimum. Furthermore, some deterministic optimization algorithms such as gradient-based algorithms require derivation of the search space, which is considered as another disadvantage of deterministic algorithms because it makes them highly inefficient in solving problems with unknown or computationally expensive derivation [69]. Moreover, deterministic algorithms assume that there is no uncertainty about the parameters of the problem. This is not the case in real problems, as it is rare to have a completely deterministic system.

In contrast to deterministic algorithms, stochastic algorithms can easily avoid local solutions through using stochastic operators. Although the use of stochastic operators might make them unreliable in obtaining a similar solution for a given problem in each run, they can avoid the entrapment in local solutions during optimization, which enables them to obtain the global optimum much easier than deterministic algorithms [70]. Besides, deterministic algorithms give a theoretical guarantee of reaching the global optimum or at least a local optimum, whereas stochastic algorithms only provide a guarantee in terms of probability because stochastic algorithms are simply a random search with some hints to guide the next potential solution to evaluate. However, despite the non-guarantee of reaching the global optimum, they are faster when compared to deterministic algorithms. Moreover, the optimization problems solved by stochastic algorithms are considered black boxes [71]. This means that they only change the inputs and monitor the outputs without the need to calculate the gradient of a solution or to know the derivation of the search space. At each step of optimization, stochastic algorithms only evaluate the solutions using the objective function and then they make decisions to improve the solutions based on the calculated objective values. This means that the process of optimization is done completely independent from the problem. Moreover, by considering the problem as a black box, stochastic algorithms become highly flexible and applicable to different types of problems [72].

Further, stochastic algorithms are broadly classified into heuristics and metaheuristics. Heuristics refer to experience-based algorithms that are designated for solving optimization problems where an exhaustive search is impractical. This means that they are used to speed up the process of finding a global optimum or an approximate solution for the cases where finding an optimal solution is difficult or when the deterministic methods fail to find the exact optimum quickly. Moreover, in contrast to metaheuristics, heuristics are problem-dependent algorithms. Meaning that they are designed to solve a specific problem without the possibility of direct application to other types of problems [73].

II.4.1 Metaheuristic Algorithms

In the last two decades, computer scientists and researchers in different fields have turned their attention toward a novel optimization paradigms known as metaheuristic optimization algorithms. These algorithms become surprisingly very popular in solving real-life challenging problems in engineering and industry. There are many reasons behind the popularity of metaheuristic algorithms such as, their efficiency and the low computational complexity compared to other existing deterministic optimization techniques. Furthermore, similar to heuristics, they are used to speed up the process of finding a global optimum or an approximate solution for cases where the deterministic methods fail to find a satisfactory solution quickly. However, there is an inappreciable distinction between heuristics and metaheuristics. In contrast to a heuristic algorithm, which is designed specifically to tackle a given problem, metaheuristics are problem-independent algorithms [74]. This means that they are used for computing the global solution or an approximate solution for many optimization problems without any special changes in the structure of the algorithm.

In addition to the above, there are many other reasons behind the interest in metaheuristics. Firstly, their structure is constructed based on very simple rules and concepts, which makes their application a very simple task for computer scientists. The simplicity of such algorithms is valuable in terms of computational complexity because it allows them to achieve a satisfactory solution in a very short period of time. Furthermore, metaheuristics are very flexible due to their high applicability to numerous optimization problems. This feature emerges from the fact that metaheuristics consider problems as black boxes. They require only the inputs which are the decision variables of the problem, and the outputs, which are the objective function values to perform the optimization task [69]. During optimization, a metaheuristic algorithm starts by creating one random input or a set of random inputs as the initial candidate solutions for the problem. Based on the number of candidate solutions, metaheuristics are classified into single solution-based algorithms and population-based algorithms. In single solution-based algorithms, only one solution is generated and improved by the optimizer. On the other hand, in population based algorithms, a number of solutions is generated, updated and improved until the best solution is found [75]. After creating the initial inputs, both types of algorithms continues the search by evaluating each solution by the objective function, observing the objective function outputs, and evolving the solutions based on their outputs until the best result is obtained or until a termination criterion is met. In general, metaheuristics are terminated when a maximum number of iterations or a maximum number of objective function evaluations is reached [76].

Last but not least, the stochastic nature of metaheuristics gives them superior abilities in avoiding local solutions compared to deterministic optimization algorithms. A set of stochastic operators assist metaheuristics to avoid search stagnation and find the real global optimum or a good approximation of the best [77].

II.4.2 Characteristics of metaheuristic algorithms

Regardless of the differences between the inspiration and the search philosophy of metaheuristics, there are still common concepts and characteristics among them, such as the division of the search process into two phases, namely exploration, and exploitation.

The exploration phase, also known as the diversification or the global search phase is an important process that every metaheuristic optimizer should have. The main purpose behind exploration is to globally investigate the search space. In this phase, the candidate solutions are encouraged to change abruptly in order to explore different regions and discover the promising areas of the available search space. The large and random changes in the solutions are the main characteristics of exploration [78]. The high rate of randomness in exploration is fruitful because it assists the optimizer in improving the diversity of the solutions and consequently discovering the promising regions of the search space.

The second phase in optimization is called exploitation or intensification. In this phase, random changes are considerably less than those in the exploration phase. The solutions tend to search locally to improve the quality and the accuracy of the best solutions [79]. Therefore, in the exploitation phase, the search is concentrated near the promising regions obtained in the exploration phase. These regions are investigated in detail hoping to obtain a good approximation of the global optimum.

As can be seen, exploration and exploitation are conflicting therefore, metaheuristics have to balance those search tendencies to avoid trapping in local optima and increase the chances of finding the global optimum of the given problem [80]. Moreover, if the metaheuristic fails to achieve a good trade-off between the two search tendencies, it will face some challenges when solving real problems. The convergence speed is one of the challenges that face metaheuristics during optimization. The term convergence describes the behavior of an algorithm towards the global optimum. According to the literature, metaheuristics suffer from what is known as premature convergence. This concept refers to the stagnation of a metaheuristic in local optima and its inability to reach the global optimum. In this case, the algorithm will converge quickly, which results in search stagnation. Usually, metaheuristics utilize sudden changes (exploration) in the solutions to avoid the stagnation in local optima [81]. Indeed, this will assist the algorithm in avoiding local solutions, however, the algorithm will not necessarily be able to converge because the abrupt changes will reduce the convergence speed towards the global optimum. Therefore, a proper balance between exploration and exploitation is a requirement to guarantee a very accurate approximation of the global optimum in a reasonable time.

II.5 Nature-inspired optimization algorithms

Nature has always been considered a rich source of inspiration for researchers. Nowadays, new algorithms are developed by observing the activities of organisms and converting them into a computational process in an optimized way. These techniques are called nature-inspired algorithms or intelligent optimization algorithms. In nature, creatures cooperate and interact in groups in order to perform many tasks including, survival, hunting, defending, and foraging. By observing the behavior of different creatures such as fish schools, ant colonies, and bird flocks, researchers realize that these creatures can find the optimal situations and perform complex tasks efficiently [82]. Therefore, a great effort is done to develop powerful optimizers by drawing inspiration from nature because it is the best and oldest optimizer on the planet. Hence, nature-inspired algorithms are designed by mimicking the natural problem-solving methods in nature, especially those used by creatures.

Further, nature-inspired algorithms are not limited to techniques inspired by animals' behavior, they further include techniques that rely on different rules observed in nature, such as the evolutionary process and physical phenomenon. For instance, physics-based algorithms usually simulate physical laws of nature, such as gravity, annealing, and thermal exchange [83]. In the world of optimization, researchers found that it is reasonable that we inspire from the different rules observed in nature to solve our problems.

Population-based metaheuristics are normally nature-inspired algorithms. Nature-inspired metaheuristics rely on several natural behaviors such as, self-organization, coevolution, and learning to create the highest quality results [84]. These algorithms can be grouped into three main categories: physics-based algorithms, biology-based algorithms or bio-inspired algorithms, and chemistry-based algorithms. Physics-based algorithms mimic the physical laws of nature such as black holes and gravity. Chemistry-based algorithms are designed based on the principles of chemical reactions such as transforming a set of reactants into products. Last but not least, the bio-inspired algorithms, which represent the big chunk of nature-inspired algorithms found in the literature, are based on the unique and successful characteristics of the biological system. In the following subsection, we will give a brief overview of bio-inspired algorithms and their intrinsic characteristics.

II.6 Bio-inspired optimization algorithms

Bio-inspired algorithms are a sub-class of nature-inspired metaheuristics. They are popularly used for optimization in almost all areas of sciences, engineering, and industries [85]. These algorithms contribute to solving optimization problems by giving the best possible solution among the set of feasible solutions in the search space. As aforementioned, the majority of the nature-inspired algorithms are inspired by the biological system. Since biology is the source of inspiration for this kind of algorithms, they are named bio-inspired optimization algorithms. These algorithms are inspired by the different principles of natural biological evolution and distributed collective of living organisms, including insects and animals.
One of the key features of biological systems is searching for the best solution using very simple rules. Ants, bees, and birds have the ability to solve complex tasks using very simple rules. These creatures evolve, self-organize, and learn in order to accomplish a common task. Bio-inspired algorithms have the ability to solve complex tasks with little or no knowledge of the search space by using the commonly shared information among individuals [84].
Bio-inspired algorithms are classified into two dominant classes: evolutionary algorithms and swarm intelligence algorithms. In the following paragraphs, we will explore the different characteristics of each class, and we will give a detailed overview of their working principles.

II.6.1 Evolutionary algorithms

Evolutionary algorithms (EA) are a subclass of evolutionary computation which belong to the higher class known as stochastic algorithms. Evolutionary algorithms are popular and old class population-based algorithms for solving different problems. EAs mimic the concepts of evolution in nature. They are inspired by the evolutionary processes observed in nature, such as reproduction, mutation, and selection [86]. EAs are used to solve multi-dimensional, nonlinear, and discrete problems without having detailed knowledge of the problems' mathematical structure.
In order to search for the global optimum, evolutionary algorithms start by creating one or more solutions in the problem search space. EAs work with a population of individuals, each of them represents a candidate solution to the problem at hand. The solutions created in the initial step are called the set of candidate solutions. Besides, they are referred to as the initial random guesses because they are created in the search space using a random operator. After creating the initial solutions, EAs attempt to find a more accurate optimal solution by itera-

tively improving the quality of the candidate solutions. The improvement is performed until the terminating condition is satisfied.
EAs provide several advantages including simplicity in use, problem independency, and local optima avoidance. On the other hand, EAs require an explicit specification of algorithmic parameters because their performance is greatly depends on the values of their parameters. In addition, EAs involve heavy computational burden [87].
Evolutionary algorithms family includes several optimization algorithms such as, Genetic Algorithm (GA), Differential Evolution (DE), Evolutionary Strategy (ES), and Evolutionary Programming (EP). In the following paragraphs, the most well-regarded algorithms in the literature namely: GA and DE, are presented in detail.

II.6.1.1 Genetic algorithm

Genetic algorithms (GA) are the most popular algorithms in the family of evolutionary algorithms. The GA was introduced to the world in 1975. The GA is a simple population-based algorithm that imitates Darwinian principles of natural evolution including selection, recombination, and mutation of genes. In GA, each candidate solution is represented as a chromosome consisting of different genes. The solution in GA is generally represented in the form of strings of binary numbers. Thus, each gene is either a one or zero. The idea of GA is simple [88]. It utilizes the survival of the fitter individuals in nature to achieve the fittest chromosome by performing reproduction, mutation, and selection operators. GA maintains the best solutions in each generation and uses them to enhance the quality of the other solutions.
In GA, the optimization is initiated by randomly creating a set of candidate solutions. Each chromosome as a candidate solution is an array of genes where each gene represents some data. The set of candidate solutions is considered as a population in GA. In the next step, the candidate solutions are evaluated by the objective function. The objective function is used as the basis for a process of selection. After evaluating the chromosomes, the best individuals are randomly selected to form a pair for creating the offspring for the next generation. Similar to the natural process, the fittest individuals in GA have a higher probability to be selected to participate in creating the next generation [89]. There are different selection mechanisms employed by researchers including, roulette wheel selection, tournament selection, rank selection, and many other selection operators.
After the selection, the chromosomes of the selected parents are combined to produce a new chromosome (solution). This process is performed by the crossover operator. In the literature, there are several techniques to perform crossover such as single-point crossover and double-point crossover. At last, one or multiple genes in the newly created population are randomly altered to mimic mutation. The mutation operator prevents the solutions to become similar and assists the GA in avoiding the entrapment in local solutions by introducing another level of randomness [90]. The three mentioned operators are used by GA to improve the population until a termination condition is satisfied. After the termination, GA returns the best chromosome in the final population as the best approximation of the global optimum. Figure II.3 shows the main stages of the GA algorithm.

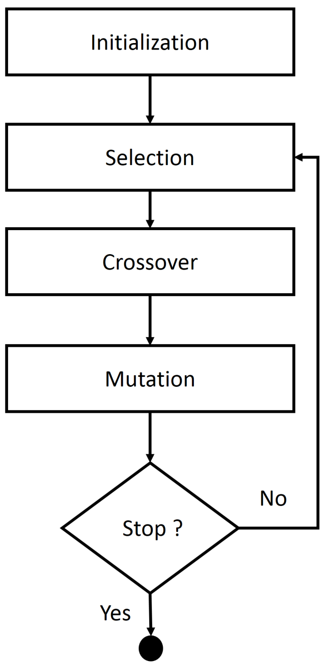


Figure II.3: GA algorithm.

II.6.1.2 Differential evolution algorithm

Differential evolution algorithm (DE) is one of the most successful types of evolutionary computing techniques. Since its emergence in 1995, DE becomes the favorite optimizer for various optimization applications in different domains of science and engineering.
The working principle of DE is similar to that used in the standard EA. DE operates through the same computational steps as employed by a standard EA. In addition, it employs difference of the parameter vectors to explore the objective function landscape [91].
Similar to GA, DE searches for a global optimum solution in a D-dimensional search space. The DE algorithm starts the optimization process with a number of random solutions being created in the search space. The candidate solutions which constitute the population are called parameter vectors or genomes. After initialization, DE performs the mutation operator in which a set of new vectors called donors are created based on a set of targets from the population. In the next step, the new offspring (solutions) are formed by recombining the target and the created donor. The recombination of the two vectors is performed through using the crossover operator. Finally, a selection is performed to determine whether the target or the offspring vector will survive through evaluating their quality using the objective function. The set of survived solutions will form the new population of the DE algorithm. The three operators are executed and the population is improved until a termination condition is satisfied. DE provides several advantages including, simplicity in implementation, faster convergence speed, few number of control parameters, and much better performance compared with other algorithms [92]. The main stages of the DE algorithm are given in Figure II.4.

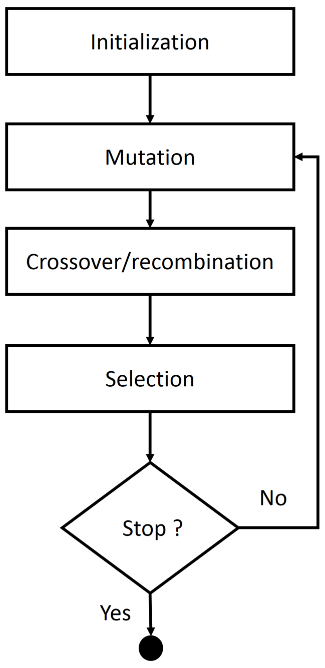


Figure II.4: DE algorithm.

II.6.2 Swarm Intelligence

Swarm intelligence (SI) is a term firstly and broadly introduced to the world of optimization in 1993 where a group of researchers used the intelligence of swarms in order to develop cellular robotic systems. Swarm Intelligence is an innovative intelligent paradigm for solving optimization problems. It is a relatively new subfield of artificial intelligence that showed a novel direction in optimization research. Swarm Intelligence refers to a family of optimization techniques that are inspired by the intelligent behavior of biological swarms [93]. Many of these swarms can be observed in nature, such as ant and bee colonies, bird flocking, and fish schooling.

It has also been observed in nature that these species can co-evolve and cooperate by interacting locally with each other and with their environment. The individuals in the swarm also referred to as search agents, tend to achieve complex goals with only simple rules and local interaction without any centralized control unit [94]. Even though these individuals only use a set of simple rules that do not exhibit sophisticated behavior when working individually, complex global optimization patterns, which are unknown to the individual agent may emerge from the interactions between them.

Below, a more formal definition of swarm intelligence is provided by Kennedy in 2006 [95]: *Swarm intelligence refers to a kind of problem-solving ability that emerges in the interactions of simple information processing units. The concept of a swarm suggests multiplicity, stochasticity, randomness, and messiness, and the concept of intelligence suggests that the problem-solving method is somehow successful. The information-processing units that compose a swarm can be animate, mechanical, computational, or mathematical; they can be insects, birds, or human* *beings; they can be array elements, robots, or standalone workstations; they can be real or imaginary. Their coupling can have a wide range of characteristics, but there must be interaction among the units.*

Such swarm intelligence has inspired researchers to develop various optimization techniques by imitating the social behavior of the different animal societies. In order to take advantage of such intelligence, researchers attempt to figure out the local rules for interactions between the individuals of a swarm that yields to this social intelligence. Swarm intelligence methods have been widely used as a suitable alternative for deterministic techniques in order to solve many optimization problems with impressive performance.
Swarm intelligence algorithms are a branch of bio-inspired algorithms inspired by the collective behavior of a population of animals. Hence, SI-based algorithms are considered population-based metaheuristics that maintain one or more populations of individuals during optimization. The main reason behind the success of swarm intelligence algorithms is that they use commonly shared information among multiple agents. Such information sharing can lead to the occurrence of self-organization, which results in a set of structures and characteristics at a higher level.

There are multiple reasons responsible for the growing popularity of SI-based algorithms, some of them are summarized below:

- SI-based algorithms are easy to implement.
- Most SI-based algorithms usually have fewer parameters to adjust.
- SI-based algorithms often use a memory to save the best solution obtained during all stages of optimization.
- SI algorithms have simple operators and they can obtain a good approximation of the global optimum in a reasonable time.

II.6.2.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a well-known swarm intelligence-based algorithm developed by Eberhart and Kennedy in 1995. It is inspired by the foraging and navigation behavior of bird flocks (Figure II.5). In nature, each bird in the swarm tends to maintain its fly direction towards the best location of the food that the swarm found so far, and the best location of food source obtained locally by him so far [96]. PSO mimics the interactions between birds in order to guide the search agents toward the best solution. In PSO, the birds are represented by particles that fly through the search space to find the optimal solution. The positions of particles represent a candidate solution to the optimization problem.
Each particle is characterized by its position X_i, a velocity V_i, and it has a memory to store information about its best local position p^{best} and the best global position found by the swarm g^{best}. During optimization, the particles explore a D-dimensional space in search for the global solution by updating their positions and velocities toward the g^{best} and p^{best} according to the following equations:

$$V_i{}^{t+1} = \omega \times V_i{}^t + c_1 \times r_1 \times (p_i{}^{best} - X_i^t) + c_2 \times r_2 \times (g^{best} - X_i^t) \tag{II.4}$$

Figure II.5: Bird flocks in nature.

$$X_i^{t+1} = X_i^{t} + V_i^{t+1} \tag{II.5}$$

Where c_1 and c_2 are acceleration constants, r_1 and r_2 are random numbers generated uniformly between 0 and 1, ω is inertia weight starts with a value 0.9 and linearly decreases to 0.4, and X_i is the position of the i-th particle in the t-th iteration.
The update process is iteratively repeated until either an acceptable g^{best} is achieved or the maximum number of iterations is reached.

II.6.2.2 Ant Colony Optimization

The ant colony optimization (ACO) is a well-regarded swarm intelligence-based algorithm initially introduced by Dorigo et al. in 1996. ACO mimics the social intelligence of ants when seeking food.
In nature, when ants search for food sources, they mark their own paths using a chemical substance called pheromone. The ants use the pheromone to communicate with other members of the colony. Upon encountering a food source, they head back to the nest while depositing pheromones in proportion to the quality of the food source [97]. The ants move and build paths incrementally by applying a local stochastic decision policy based on the use of pheromone traces.
When other ants encounter the pheromone marks of an ant, they follow the ant's path and leave their own paths if the level of pheromone in the discovered path is higher. If the food source is discovered by multiple ants, each ant will build a different path to the food source. It is observed that eventually, all ants tend to choose the shortest path to the food source [98]. This is achieved using the pheromone marks because the pheromone vaporizes with higher rate before it is re-marked by other ants in longer paths. Whereas in shorter paths, the level of pheromone is high, which makes the ants follow it and abandon the paths with weaker pheromone levels. Using

this mechanism, the shortest path to the food source is always selected by ants, as illustrated in Figure II.6.

Figure II.6: Ants' path in nature.

The described behavior of ants is the main inspiration of the ACO algorithm. In ACO, at each step, the ant k which is on the point i applies a probabilistic transition rule to select which point j it will visit next. The transition probability of the ant from point i to point j is given by:

$$p_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})} \tag{II.6}$$

Where the variable $\tau_{i,j}$ is the intensity of the pheromone on the edge (i, j), $\eta_{i,j}$ is the heuristic value of the path from point i to point j. α and β are two parameters that determine the relative influence of the pheromone and the heuristic on the decision of the ant.
The intensity of pheromone traces decreases over time by a constant factor called evaporation coefficient. From a practical point of view, pheromone evaporation is necessary to avoid premature convergence of the algorithm. After each step, the pheromone intensity of each edge (i, j) is updated according to the equation below:

$$\tau_{i,j} = (1 - \rho)\tau_{i,j} + \Delta\tau_{i,j} \tag{II.7}$$

ρ is the evaporation coefficient of the pheromone in the range $(0, 1)$ and the amount of pheromone added by the ant is given by:

$$\Delta\tau_{i,j} = \sum_{k=1}^{m} \Delta\tau_{i,j}^{k} \tag{II.8}$$

Where m is the number of ants in the colony and $\Delta\tau_{i,j}^{k}$ indicates the amount of pheromone deposited by the k-th ant.
The ACO is used intensively in the literature to optimize several problems. It's most popular use is solving the well-known travelling salesman problem.

II.6.2.3 Artificial Bee Colony Algorithm

The Artificial Bee Colony (ABC) algorithm was introduced by Karaboga in 2005 for solving optimization problems. It is a swarm intelligence-based algorithm that simulates the foraging behavior of honeybees (Figure II.7). ABC is a population-based optimization algorithm, in which the colony is divided into three groups of bees: employed bees, onlookers, and scouts. Employed bees exploit food sources and examine the properties of the discovered food sources such as the quantity of nectar, direction, the distance between food sources and the hive and they carry this information back to the hive and share it with onlooker bees [99]. Each of the employed bees is associated with one and only one food source, so the number of employed bees is equal to the number of food sources around the hive. Recruitment of unemployed bees for foraging is the crucial part of the algorithm. The onlookers wait in the hive to select good food sources to exploit depending on the information shared by the employed bees. The information exchange happens in the dancing area of the hive using a special dance called the "waggle dance".

The nature of dance is proportional to the nectar content of the food source exploited by the dancing bee. Therefore, good food sources attract more onlooker bees. The employed bees whose food sources are exhausted become scout bees [100]. These bees abandon their food sources and carry out a random search for new ones.

The ABC has been successfully applied for solving several combinatorial and discrete optimization problems such as feature selection, multicast routing, and machine scheduling.

Figure II.7: Honeybees in nature.

II.6.2.4 Other swarm intelligence algorithms

The literature provides a significant number of swarm intelligence algorithms. The number of the algorithms is high to the extent that it is not possible to list all the existing algorithms in one table. Therefore, the most popular swarm intelligence algorithms found in the literature are listed in Table II.1.

Table II.1: Swarm intelligence algorithms.

Algorithm	**Publication Year**
The Aquila optimizer	2021
The Archerfish Hunting Optimizer	2021
Golden eagle optimizer	2021
Artificial lizard search optimization	2021
Black widow optimization algorithm	2020
Bald eagle optimizer	2020
Water strider algorithm	2020
Tunicate swarm algorithm	2020
Rat Swarm Optimizer	2020
The Sailfish Optimizer	2019
Seagull optimization algorithm	2019
Harris hawks optimization	2019
Emperor penguin optimizer	2018
Squirrel search algorithm	2018
Earthworm optimization algorithm	2018
Owl search algorithm	2018
Grasshopper Optimization Algorithm	2017
Salp Swarm Algorithm	2017
Spotted hyena optimizer	2017
The Whale Optimization Algorithm	2016
Crow search algorithm	2016
Dolphin Swarm Optimization Algorithm	2016
The Ant Lion Optimizer	2015
Moth-flame optimization algorithm	2015
Dragonfly algorithm	2015
Elephant herding optimization	2015
Grey Wolf Optimizer	2014
Social spider optimizer	2013
Dolphin echolocation algorithm	2013
Krill herd optimizer	2012
Bat algorithm	2010
Glowworm swarm optimization	2009
Cuckoo search	2009
Firefly Algorithm	2008
Fish swarm algorithm	2002

II.7 Conclusion

In this chapter, we have discussed the optimization methods, their classification, and we have highlighted the different characteristics of each class. In the literature, a significant number of optimization methods are proposed to solve optimization problems. Researchers have focused on important novel ideas for solving these problems with the minimum cost.
Deterministic methods are very popular, and they are still being used by different researchers for optimization. For some problems, the use of deterministic methods is necessary. However, it is not always possible to apply these methods due to their shortcomings. Deterministic methods have some disadvantages where they fail to provide satisfactory results in an acceptable time when the complexity and the problem dimension are high.
As an alternative, bio-inspired metaheuristics have emerged, and they have already proven their efficiency in solving several challenging problems. The intrinsic characteristics of these algorithms, such as simplicity, applicability, and reasonable computational time, make them well suited for solving different optimization problems.
Another feature provided by bio-inspired algorithms is the possibility of combining the strengths of two or more algorithms in order to construct an efficient optimization method. This mechanism is known as hybridization. Using hybridization, the weakness of an algorithm is compensated by another algorithm and vice versa. Hybridization is becoming a trend in solving combinatorial optimization problems in recent years.

CHAPTER III

DEPLOYMENT OF WIRELESS SENSOR NETWORKS: A STATE OF THE ART REVIEW

Contents

III.1 Introduction

Deployment optimization is a crucial issue that must be taken into consideration while designing an efficient wireless sensor network. Similar to various problems related to the design of WSNs, deployment optimization got its share of attention from the research community. In the literature, several studies have proposed plenty of solutions that differ according to the optimization objective, the method, and the application where the WSN is implemented.
This chapter is designed to give a detailed survey that focuses on the deployment problem in WSNs. Firstly, we present a general overview of the deployment problem in WSNs. Secondly, we highlight the most important objectives, which have a great impact on the performance of WSNs including, coverage, connectivity, cost, and network lifetime. Thirdly, we present a complete and up-to-date review of the deployment techniques of WSNs. The different deployment techniques are studied and classified into several categories. We categorize the deployment techniques based on the method used to obtain the optimal solution. Throughout this review, we offer a systematic overview of the existing deployment works, and we highlight the advantages and disadvantages of each work. Finally, we will end the chapter with a set of conclusions about the current scope of WSNs' deployment.

III.2 The deployment problem in WSNs

The deployment problem is a critical issue that directly affects the intrinsic performance criteria of WSNs. Successful operation of WSN highly depends on the positions of sensors. Based on the application, sensors are deployed to achieve one or many objectives. One common objective in many applications, which is considered the main purpose behind the usage of WSNs, is coverage [101]. Depending on the WSN's usage, whether it is for point surveillance, target tracking, or area surveillance, the goal is to place the sensors in optimal locations to achieve the maximum possible coverage and ensures a complete collection of data [102]. Furthermore, ensuring that the required coverage can be achieved with a fewer number of sensors is another objective of deployment. By deploying a smaller number of sensors, the network becomes less crowded, and several problems can be avoided, such as coverage redundancy and communication overhead. Moreover, using the appropriate deployment plan, the complexity of several problems in wireless sensor networks can be reduced, such as routing and energy conservation. Above all of this, all the deployment objectives have to be achieved using techniques that allow the saving of sensors' energy. Therefore, energy efficiency is of great concern in the deployment of WSNs.
The complexity of the deployment problem arises from the constrained nature of sensors and the different application requirements. So there is no deployment technique capable to satisfy all usage requirements as several objectives are usually conflicting, such as maximizing the coverage and minimizing the cost and energy consumption [103]. Besides, achieving multiple design objectives such as maximum coverage with the minimum cost puts the deployment problem in the category of the NP-hard problems. Furthermore, optimizing conflicting objectives adds an additional layer of complexity to the deployment problem, which makes the choice of the

proper deployment technique a very critical task.
Different studies in the literature have given the main characteristics and design steps that any deployment technique must have. Some of them divided the deployment into three phases: pre-deployment and deployment phase, post-deployment phase, and re-deployment phase. In the first phase, the sensors are placed manually or launched from the air over the area of concern. The post-deployment consists of adapting the wireless sensor network in case the network topology is changed. For the re-deployment, it is needed when a set of sensors are added or removed from the network [104]. The other studies that are interested in maximizing the amount of gathered data have addressed the deployment problem from another perspective where they divide the deployment into two main phases namely coverage and placement. In the coverage phase, a large number of sensors are placed such that all the targets are covered. The placement has the objective of minimizing the number of the used sensors while maintaining the same coverage quality [105]. However, these design steps are not feasible for all situations and applications because the deployment problem depends to a large extent on the application concerns, the characteristics of sensors, and the deployment environment. For instance, when the area of concern is inaccessible, the placement phase is impossible to perform. Therefore, finding a good deployment technique that addresses all the application concerns is considered a challenging task in the researches of the deployment problem.

III.3 Deployment modes

The deployment strategy is used to define the topology of the network, the number, and the position of sensor nodes. Different modes are proposed in the literature to deploy WSNs. The choice of the appropriate mode mainly depends on the application and the environment where the sensors are deployed. Figure III.1 shows the two popular deployment modes: static (deterministic) mode and dynamic (random) mode [108].

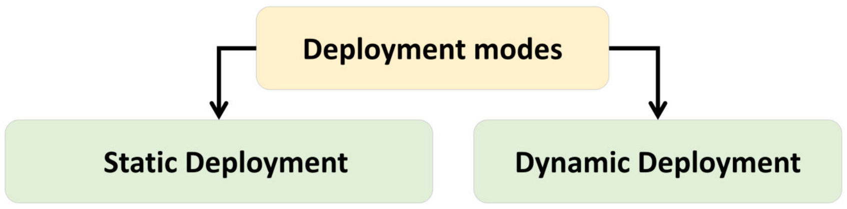


Figure III.1: Deployment modes.

III.3.1 Static deployment

In static deployment, the sensors are precisely placed in exact locations to achieve the required coverage and connectivity. Besides, static WSNs are easy to implement. The locations of static sensors are precisely chosen based on a predefined deterministic pattern. These locations remain constant during the network lifetime. Static deployment is generally used when deploying the WSN in a friendly environment.

III.3.2 Dynamic deployment

In contrast to static deployment, dynamic deployment is used when the sensing area is inaccessible or when the deployment is scaling over a large geographical region. In these circumstances, the only available choice is random deployment, in which the sensors are scattered from the air over the sensing area. Due to this random scattering, some areas can be highly covered whereas others are not covered at all resulting in what is known as coverage holes. Therefore, the random deployment of sensors might not always meet the desired requirements. Generally, in the case of random deployment, most applications utilize mobile sensors, which gives them the ability to improve the coverage by moving the sensors to new locations after the initial random scattering.

III.4 WSN deployment objectives

The following sections outline the different WSN deployment objectives and highlight their importance for designing an efficient deployment scheme. These objectives include coverage, connectivity, energy efficiency, network lifetime, and cost.

III.4.1 Coverage

Coverage is considered an important criterion when evaluating the effectiveness of a WSN. When deploying WSNs, a set of sensors is placed in a region of interest. Each sensor has a limited sensing range, and it can detect only events and objects within its sensing range. The sensors are supposed to cover all the objects located at different places inside the region of interest for better performance [106]. Therefore, the quality of surveillance, which the network can offer is determined based on the coverage quality. Coverage can be defined as how effectively the WSN can monitor areas, points, and objects. Maximizing the coverage becomes a basic requirement that directly affects the sensor network quality of service. Therefore, almost all applications consider coverage as the main objective during deployment.
Based on the application and the stipulation of monitoring of events, we distinguish three types of coverage: area coverage or blanket coverage, point coverage, and barrier coverage.
A more detailed classification is provided in Figure III.2. Coverage type can be referred to as θ-coverage, where θ refers to the percentage of the covered area. From that perspective, coverage type can be classified into full coverage, where θ=1, and partial coverage, where θ ranges from 0 to 1. Partial coverage can be further divided into point, path, and trap coverage. Point coverage is where coverage of a specific point is required, which can be further classified into focused coverage, and target coverage. Path coverage; on the other hand, requires a certain path in the deployment area to be covered. It can be classified into barrier, and sweep coverage [107].

III.4.1.1 Area coverage

Area coverage aims to provide continuous surveillance of all the points and objects located inside the region of interest. An object in the area of interest is said to be covered if it

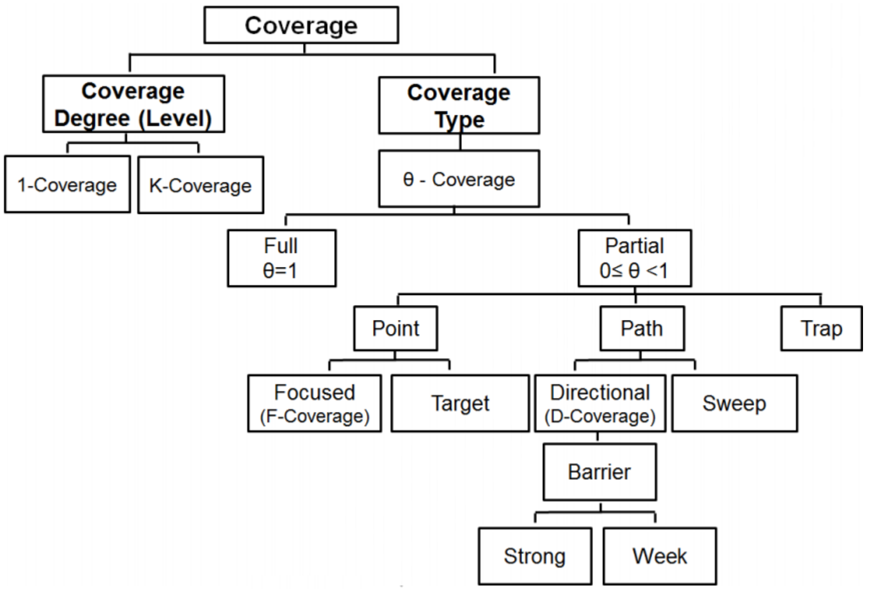


Figure III.2: Coverage types.

is detected by at least one sensor node [107]. According to application requirements, the required coverage can be full or partial. Deployment techniques generally attempt to achieve full coverage. However, in some situations, the number of sensors is limited, which prevents the WSN from providing full coverage. Therefore, the goal becomes to provide partial coverage where only the important regions and objects are covered.

Full coverage

The main demand of many applications is to obtain the best quality of surveillance from the WSN. WSN can provide it by ensuring that every location in the entire area is covered by at least one sensor node; this type of coverage is referred to as 1-coverage. The surveillance area can be a region of any shape with various conditions. If each point is covered with at least one of the deployed sensors, effective data collection about all the events that take place inside the area of interest is guaranteed. As an example, the military requires full coverage of battlefield areas in order to detect intruders and make critical and effective decisions. In some other applications, covering each point with only one sensor is not enough due to several reasons, such as when the sensors are deployed to track important objects. Such applications require that at least k ($k > 1$) different sensors to cover each point in the area of interest; this type of coverage is called k-coverage [108].

Partial coverage

Partial coverage is an objective of applications that do not consider full area coverage a necessary requirement or do not afford the high complexity and cost of full coverage. This kind of application requires only an acceptable coverage quality, which is generally expressed as a percentage. In partial coverage, sensors cover only a percentage of targets. In contrast, 100% coverage is required for full coverage. Thus, full coverage can be regarded as a special case of partial coverage [109]. Similar to full coverage, in partial coverage, either 1-coverage or k-coverage of the targets is needed. Partial coverage requires a small number of sensors to achieve the desired purpose. Therefore, it helps to save the network cost and the energy of sensors. The applications of partial coverage are numerous, for example, temperature measurement systems require to sense the temperature of 70% or 80% of the area of interest in order to estimate the temperature in the whole area.

III.4.1.2 Point coverage

Point coverage, also called target coverage, is considered a requirement in many applications. Point coverage aims to monitor a set of discrete target points located inside the region of interest. The target points are usually given prior to the deployment of sensors. The nearby sensors have to detect all the events that happen in the defined targets and report the data back to the sink for processing [102]. Based on the application and the nature of targets, the point coverage can be classified into fixed point coverage and mobile point coverage.

Fixed point coverage
A target point is said to be fixed if it always maintains the same location. The fixed points are determined prior to the deployment of WSNs. The monitoring of fixed points is easy, and only static sensors can be used to cover them [110]. When monitoring fixed points, the sensors are precisely placed at accurate locations inside the area of interest to allow real-time surveillance of the targets. Monitoring enemy bases and spots of exchange are popular examples of fixed point coverage.

Mobile point coverage
A mobile target point changes its location frequently and abruptly. When covering the mobile target point there are two possibilities, if static sensors are deployed, they must be placed in strategic locations so that for each new position of the mobile target, there will exist at least one sensor that can cover it. The second possibility and the most effective is to use mobile sensors. Firstly, the mobile sensors have to be deployed in locations that allow them to cover the initial position of the moving target. Secondly, the sensors are required to follow the moving target as it moves to new positions, and at least one sensor has to cover the point in a given time in order to avoid the loss of the target [111].

III.4.1.3 Barrier coverage

Barrier coverage is used by applications that aim to detect intruders passing across borders. The sensors are deployed along an open or closed belt shape region to detect objects that attempt to penetrate the area of interest. For efficient monitoring, each and every point of the border should be covered by at least one sensor. Barrier coverage assures the detection of all types of intruders regardless of their penetrating direction and their speed [112]. Barrier coverage is mainly used to prevent illegal breaches of international borders and to detect potential sabotage of all types of pipelines. Depending on the requirements and circumstances of applications, barrier coverage is classified into strong barrier coverage and weak barrier coverage [113]. The first assures efficient monitoring since it can detect all border intruders whatever path they take. Therefore, the path followed by the object does not matter. The latter can detect only objects moving along congruent paths.

III.4.2 Connectivity

Many WSN applications expect the guarantee of network connectivity during and after deployment. The significance of connectivity cannot be neglected and should have the same degree of importance as coverage. Ensuring full connectivity is necessary to guarantee the transfer of the gathered data to the sink for processing [114]. Usually, connectivity is defined as the ability of each sensor to find a direct or a multi-hop path to reach the sink. A sensor node is said to be connected if its location is within the communication range of at least one connected node. A WSN is often represented by a graph with the sensors as vertices and the communication link between a pair of sensors as an edge. A WSN is fully connected if the associated graph is connected [115]. Due to the limited energy, a sensor can send data only for short distances. Therefore, in multi-hop communication, the sensors exchange data with their direct neighbors until it reaches the sink. In order to ensure reliable data transmission, the deployment technique has to prevent the formation of disjoint groups of sensors by organizing the sensors into a connected network.
In most applications, ensuring that each node has exactly one neighbor directly connected to it is sufficient for reliable data transmission. This type of connectivity is called 1-connectivity. However, in applications that operate in harsh environments where it is very likely that sensor failure will occur, ensuring that each sensor is connected to at least k ($k > 1$) sensors is a basic requirement. The k-connectivity ensures the continuous operation of WSN despite the failure of one or more sensors [116]. Furthermore, k-connectivity is considered an efficient solution that ensures fault tolerance and improves the WSN resistance in case the energy is fully depleted.

III.4.3 Network lifetime

WSNs are usually deployed in inaccessible environments, and with the limited battery capacity of sensors, their lifetime is considered a major design objective during deployment. Unless they have a direct power supply, which is not the case in inaccessible environments, generally, sensors are powered using limited batteries. In inaccessible environments, it is difficult to recharge or replace the batteries, which means that it will not take a long time before their energy is depleted and the sensors become nonfunctional [117].
When reviewing the literature, we can find several definitions that have been proposed for the WSN lifetime. Each definition is originated from a different perspective based on the specific application and the objective function of the WSN. However, there is a general definition of the network lifetime that summarizes all the definitions: we say that the network lifetime is expired when the network degrades to a point, in which it is no longer able to perform its intended function [118]. That point could be the time instant at which a certain number of nodes in the network depleted their batteries or when the network is partitioned, and the communication between the disjoint groups is lost, or when coverage is degraded below a predefined threshold. Controlling energy consumption during deployment has a significant influence on the lifespan of

the WSN. If sensors are uniformly deployed, some sensors, which are located near the sink will deplete their energy faster than the others in the network. This situation occurs frequently in WSNs because all the incoming communication to the sink passes through these sensors. The extensive data transmission performed by these sensors will deplete their energy more quickly, and eventually, they will die [119]. Due to this, the network performance will degrade, and the connectivity with the sink will be lost. Therefore, optimal placement of sensors is required to balance the energy consumption of sensors and prolong the network lifetime.

III.4.4 Network cost

From a user point of view, one of the major objectives in the deployment of WSN is cost. Achieving a good coverage quality and guaranteeing a long lifespan of the WSN at a minimum cost are basic requirements for the users. The cost of WSN is directly related to the number and the type of sensors. Deploying a large number of sensor nodes leads to a high deployment cost [120]. Therefore, many applications aim to achieve the maximum possible coverage with the least number of used sensors. By deploying a smaller number of sensors, several problems can be avoided, such as coverage redundancy and communication overhead. Furthermore, the deployment of a smaller number of sensors reduces the WSN's deployment cost because deploying fewer sensors decreases the monetary expenditure of the user.
When deploying sensors of the same type, all sensors will have the same cost. Therefore, deployment cost can be minimized by achieving the lowest sensor count with the satisfaction of the user's requirements. However, when deploying a different type of sensors, the cost can be minimized by finding the best possible combination of sensors that can satisfy the requirements because the cost varies according to the sensor type [121]. In practical applications, rather than using expensive sensors with high characteristics, we can construct a mixed sensor network composed of different kinds of sensors to achieve a balance between performance and cost.

III.5 WSNs deployment techniques

Finding optimal solutions to the deployment problem is considered a very active search field. The literature provides a huge number of methods that differ according to the optimization objective, the method, and the application. In the rest of this chapter, the different deployment strategies including grid-based techniques, geometry-based techniques, virtual force-based techniques, integer linear programming techniques, and bio-inspired deployment techniques are discussed in detail. Figure III.3 illustrates a multidimensional classification of deployment techniques.

III.5.1 Grid-based deployment strategies

Grid-based strategies are used to deploy the sensors deterministically in the area of interest. In this type of deployment, the area of interest is divided into virtual grid cells having the same size. Figure III.4 shows that several cell shapes are used to represent the area of a cell, such as triangular shape, square shape, and hexagonal shape. After the partitioning of the area, sensors

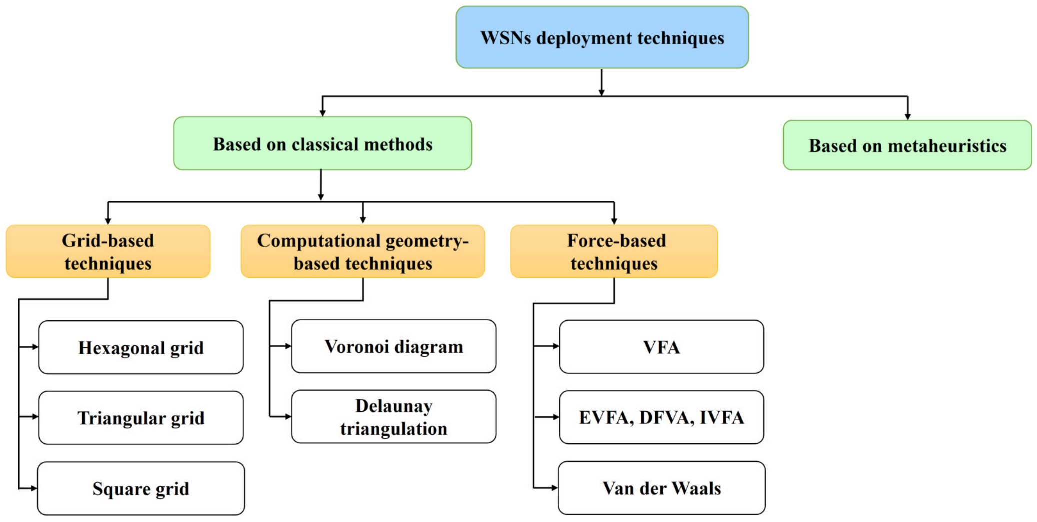


Figure III.3: Deployment techniques classification.

are precisely placed in optimal locations to achieve the specific application requirements [122]. Sensors are either placed on the center of the cells or at the cell vertices, in which their sensing zones overlap. It is favorable to use the regular patterns to deploy the sensors because such patterns provide an acceptable degree of coverage and connectivity. Furthermore, grid-based deployment strategies allow the application to achieve several levels of coverage and connectivity using a few sensors only by adjusting the relationship between the communication and the sensing ranges.

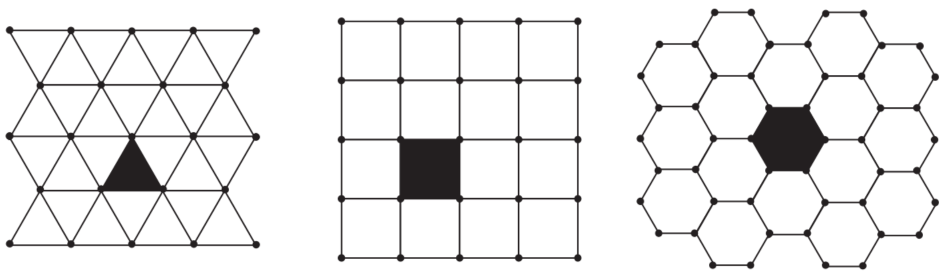


Figure III.4: Regular patterns for deployment.

The work in [123] presented a grid-based strategy for deterministic deployment of sensors. In the proposed strategy, the sensing area is divided into equal square cells. Each sensor node is supposed to occupy a cell center to cover it. After initial random deployment, the sensors use the location information of their neighbors to locate the uncovered cells. After covering the located cells, the sensors attempt to maximize the coverage by covering the cells that do not have any sensor. Therefore, if the sensing range of the sensor allows him to cover the neighboring cell and maintain the coverage of its current cell, it should decide where to move to cover

the two cells. The sensors perform these steps until all cells are covered by at least one sensor.
In [124], a hexagonal grid-based deployment algorithm called Push & Pull is proposed. Push & Pull is a distributed algorithm where the decisions regarding the behavior of each sensor are taken locally using the available information. Push & Pull starts by dividing the area of interest into a number of hexagons, where the hexagon side length equals the sensing range of sensors. The basic idea of Push & Pull is as follows: when a sensor is positioned at the center of a hexagon, it examines the hexagon area looking for other sensors located within its sensing range. After the neighbor discovery, the sensor determines whether the adjacent hexagons are still to be covered. If there are uncovered neighboring hexagons, the sensor leads the sensors located inside its hexagon to cover the adjacent hexagons. After leading all the neighbors, if some hexagons are left uncovered, the sensor initiates the pull operator to attract other sensors towards the uncovered hexagons. Push & Pull builds progressively the hexagonal cells until all the sensors are deployed. Push & Pull is capable to achieve complete coverage and guarantee connectivity when the relation between the transmission radius R_t and the sensing radius R_s is $R_t \geq \sqrt{3R_s}$.
In [125], another grid-based deployment strategy that minimizes both the cost and energy consumption is proposed. A regular hexagonal architecture is used to divide the area of interest into equal cells in order to satisfy the constraints of coverage and connectivity. The hexagonal cells are further divided into groups, each of which represents a layer. The center hexagon of the area is considered the first layer, and the cells located at the boundaries of the area are considered at the final layer. At first, the sink is placed in the first layer, and then the algorithm uses a set of theorems and constraints to optimally place the sensors in the other layers. Sensors are deployed at the center of hexagons to monitor the hexagon area. The sensors are placed in cells such that the cost is minimized and the energy consumption of sensors in the same layer is balanced. After a set of experiments, the authors show that their algorithm can reduce the network cost by uniformly deploying the sensor nodes in appropriate cells.
The triangular deployment pattern is used in [126] to maximize the coverage using the minimum number of sensor nodes. The authors proposed a centralized deployment algorithm named HGSDA to deploy the sensors in a triangular lattice. Similar to the above-described techniques, HGSDA divides the area of interest into hexagonal cells where the center of the cell represents a candidate position for a sensor. Although the cells have a hexagonal shape, HGSDA deploys the sensors in a triangular lattice by ensuring that the distance between every two neighbors is $\sqrt{R_t}$. The proposed algorithm starts by identifying the redundant sensors that cover the same cell, and then these redundant sensors are moved to empty cells to maximize the coverage. The results indicate that HGSDA can guarantee full coverage if a sufficient number of sensors are deployed.
To achieve high degrees of coverage and connectivity, authors in [127] study a set of deployment techniques based on the triangular, square, and hexagon patterns. The goal of the three deployment techniques is to achieve p-coverage and q-connectivity with different ratios between the sensing and communication ranges (where p and q are smaller than 6). Each technique attempts to form an optimal regular deployment pattern by deploying a minimum number of sensors that achieve the required coverage and connectivity levels. The authors presented sev-

eral theorems to adjust the distance between the sensors and estimate the number of sensors required to achieve the predefined coverage level. Besides, the authors conducted a comparison between the three patterns, and they recommended that the triangular pattern is better than the other two especially, when $p > 3$. Table III.1 shows a comparison between grid-based deployment techniques.

Table III.1: A comparison between grid-based deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[123]	✓				-Achieve full area coverage -Coverage expansion speed is high	-Increase the moving distance of sensors
[124]	✓				-Increase coverage even in areas of irregular shapes -Autonomous deployment of mobile sensors	-High energy consumption -High communication overhead due to the frequent transmissions
[125]	✓			✓	-Balance the energy Consumption -Reduce the unnecessary energy costs during deployment	-The configuration of initial energy is not arbitrary
[126]	✓				-Achieve a complete coverage -Reduce sensor node moving distance	-High processing time

[127]	✓	✓			-Provide multiple degrees of coverage and connectivity	-Different ratios of the communication range to the sensing range degrade the performance and affect the coverage

III.5.2 Computational geometry-based deployment strategies

Computational geometry strategies are another class of methods used for the deployment of WSNs. These strategies are based on geometrical objects such as points, polygons, and line segments. In general, computational geometry strategies are used to eliminate coverage holes by relocating mobile sensors from densely deployed areas to sparse ones. Voronoi diagram and Delaunay triangulation are considered the utmost well-liked computational geometry techniques used for WSN deployment.

III.5.2.1 Voronoi diagram

The Voronoi diagram is an important structure in computational geometry that represents the proximity information about a set of geometric nodes. In Voronoi diagram, the region of interest is partitioned into a number of sub-regions called Voronoi polygons [128]. In WSNs, the partition is performed based on distances between sensor nodes. Each Voronoi polygon is occupied by only one sensor node such that every point in the polygon is closer to the sensor node in this polygon than to any other node. Each polygon is composed of several edges called Voronoi edges, where the intersection point of two or more Voronoi edges is called a Voronoi vertex. Two sensors are neighbors if their Voronoi polygons share one Voronoi edge. The Voronoi diagram for a WSN is constructed based on the initial positions of sensors, then if there are uncovered areas within the polygons, sensors are moved to cover them. Figure III.5 represents the Voronoi diagram.

In [129], the authors presented three deployment algorithms based on Voronoi diagram, namely, the vector-based algorithm (VEC), the Voronoi-based algorithm (VOR), and Mini-max. The idea behind the three algorithms is to detect and determine the existence of coverage holes using Voronoi diagram. In all the algorithms, the area of interest is partitioned using Voronoi diagram, and sensors are placed to effectively cover Voronoi cells. In VOR, if a hole is detected, the sensors calculate where to move to eliminate or reduce the size of that hole. Min-max is similar to VOR, however, the sensors are pushed to cover the furthest Voronoi vertex in order to eliminate coverage holes. The characteristic that distinguishes Min-max is that sensors move more conservatively with the consideration of not generating new holes. Whereas in VEC, the

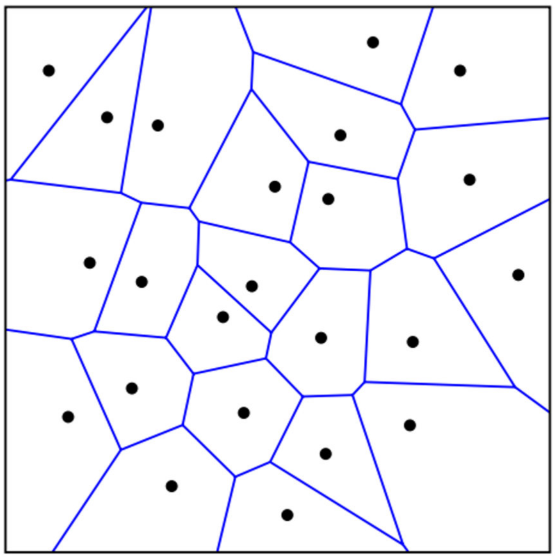


Figure III.5: Voronoi diagram.

behavior of electromagnetic particles is used to move sensors away from dense areas, where sensors are placed close to each other.
Based on the multiplicatively weighted Voronoi (MW-Voronoi) diagram, three deployment algorithms are developed in [130] to solve the coverage problem in mobile sensor networks. The algorithms are abbreviated as MWV, MWP, and MDW. The authors assume that the deployed sensors have different sensing capabilities, and the area of interest is free from any obstacles. The goal of the three algorithms is to detect coverage holes in the MW-Voronoi region of each sensor and then move the sensor to an optimal location to reduce the coverage hole. Therefore, each sensor is required to sweep its MW-Voronoi region looking for a coverage hole. If a hole is identified, the sensor moves toward a point in its MWVoronoi region, which has the maximum coverage weight. In MWV, the sensor moves toward the vertex with maximum weight in its MW-Voronoi region. In MWP, the sensor chooses a point in its MWVoronoi region that has the maximum weight and starts moving toward that point. Unlike MWV and MWP, the sensor in MDW chooses its destination based on both distance and weight because MWV and MWP do not move sensors when the weight of all points of the area are equal.

III.5.2.2 Delaunay triangulation

In graph theory, Delaunay triangulation is formed based on Voronoi polygons, and it is considered the dual graph of the Voronoi diagram. Delaunay triangulation is used to obtain the two nearest neighboring sites by taking the shortest edge in triangulation. We can construct Delaunay triangulation by connecting every two adjacent points in the Voronoi diagram whose polygons share a common edge. By connecting the adjacent points, the area is partitioned into triangles so that no point in any triangle is located within the circumscribed circle of any other triangle in the area. The empty circle property of the triangulation is used in WSNs to eliminate the coverage holes. Figure III.6 illustrates the Delaunay triangulation constructed from the Voronoi diagram shown in Figure III.5.

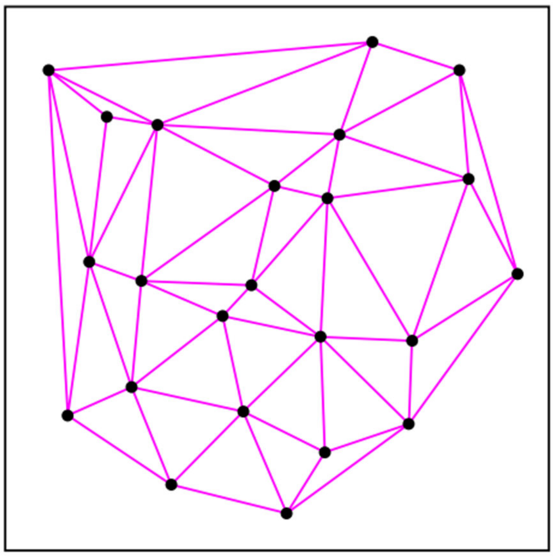


Figure III.6: Delaunay triangulation.

A deterministic deployment algorithm based on computational geometry is proposed in [131] to maximize the coverage of an area that contains obstacles with regular and irregular shapes. The first step of the proposed algorithm is to deploy the sensors over the entire area, even over obstacles. The second step is to minimize the number of deployed sensors by eliminating the sensors that are located inside the obstacles. When eliminating sensors, coverage holes with different sizes may occur around the obstacles. Therefore, the role of the algorithm is to detect these coverage holes and then partitioning the holes into triangulations using Delaunay triangulation. Finally, sensors are placed in the triangle vertices to cover the holes. To ensure full coverage, the length of triangles' edges are set to be less than the sensing range.
In [132], a centralized deployment technique called DT-Score is proposed for solving the coverage problem in an area with obstacles. To optimize the deployment, this technique attempts to cover the sparse regions in the area of interest. In order to achieve this, two deployment phases are presented. In the first phase, a contour-based deployment method is used to detect and eliminate the coverage holes near the edges of the area and obstacles. The second phase employs the Delaunay Triangulation to deploy the sensors in optimal positions for covering the remaining regions in the area of interest. The empty circle property of the Delaunay triangulation is used to find the sparse regions and place the sensors in locations that achieve the most coverage gain. Table III.2 shows a comparison between the previously mentioned computational geometry-based techniques.

Table III.2: A comparison between computational geometry-based deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[129]	✓				-Maximize the area coverage through reducing coverage holes -They have the ability to deal with obstacles	-High computational complexity for coverage holes detection -Require complex calculation when estimating new positions of the sensors
[130]	✓				-Improve the initial coverage of mobile sensors -Support the deployment of sensors having different sensing capabilities	-High computational complexity -High mobility
[131]	✓		✓		-Achieve full area coverage with minimum number of sensors -Deal with regions having arbitrary boundaries and obstacles	-High computational complexity for coverage holes detection

[132]	✓				-Achieve high coverage in an area with obstacles -It is scalable and support different types of sensors	-Increase the network cost -The locations of obstacles must be known before deployment

III.5.3 Force-based deployment strategies

In the virtual force-based deployment technique, the locations of sensor nodes are determined on the basis of the forces that are originated from the interactions among them. Different virtual forces including, attraction, repulsion, and friction, affect the sensor nodes. For optimal placement, a sensor node should maintain a fixed threshold distance that separates it from others. Generally, the threshold distance is equal to the sensing range of the sensors. Each sensor affects others by exerting a repulsive force if the distance separating them is less than the fixed threshold. On the other hand, the attractive force is applied if the distance separating two neighboring nodes is greater than the fixed threshold. When the distance is equal to the fixed threshold, neither attraction nor repulsion is applied because the exerted force is null. The working principle of virtual force algorithms is illustrated in Figure III.7. As can be seen from the figure, the position of sensor 3 is changed according to the forces applied by sensor 1 and sensor 2.

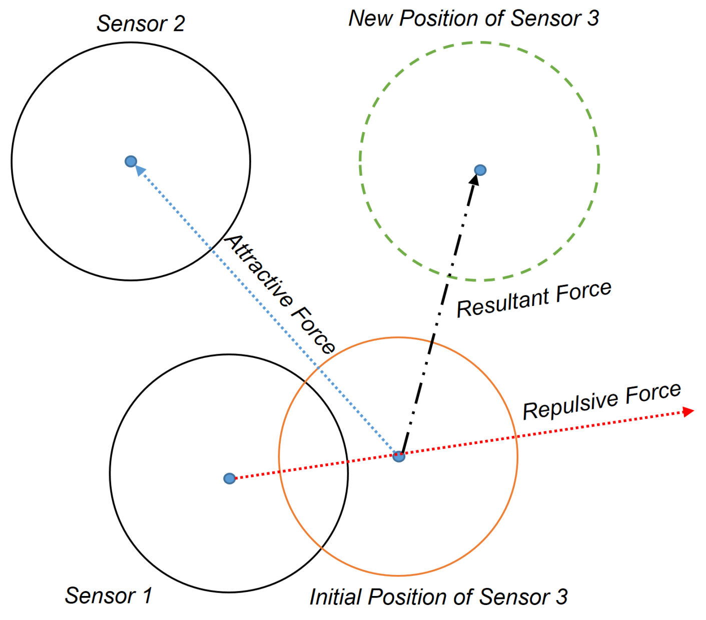


Figure III.7: Virtual force working principle [111].

The virtual force algorithm (VFA) is introduced in [133] to tackle the problem of area coverage in WSNs. VFA is a centralized algorithm that aims to improve the initial coverage of randomly deployed sensors. After the random placement, sensors experience three varieties of forces in order to relocate them into optimal positions. The uncovered regions, obstacles, and sensors themselves exert a combination of attractive and repulsive forces to determine new locations for sensors that satisfy coverage requirements. If the area contains obstacles, it will exert a repulsive force on sensors to push the sensors away. The regions of preferential coverage exert an attractive force to attract the sensors toward uncovered area blocks. At last, depending on the orientation and distance between the sensors, they exert either repulsive or attractive forces on each other. Therefore, based on the above description, the net force on a sensor node can be calculated by summing the three types of forces. As mentioned above, VFA is a centralized algorithm, meaning that it is executed by the sink node and the sensors do not change their positions until the final positions are determined by the sink node. Sensors receive their final positions from the sink, and they relocate themselves in these positions to enhance the network coverage. VFA does not work properly when network connectivity is not initially ensured.
To overcome the connectivity maintenance problem in VFA, an Extended Virtual Forces-based (EVFA) technique for the self-deployment of WSNs is proposed in [134]. The authors discussed that the connectivity maintenance problem occurs when the communication range is low. Therefore, they introduce the orientation force to improve the force model and helps the EVFA to keep the continuous connectivity during sensor movement and eliminate coverage holes. Besides, EVFA was designed to solve the node stacking problem, in which two or more sensors occupy almost the same position. This problem is encountered in the VFA when the attractive forces coefficient is not well-tuned, and the ratio of the communication range to the sensing range is large. To alleviate this problem, a novel exponential force model is introduced in EVFA to adjust the distance between a sensor and its faraway neighbors because attractive forces between a sensor and its faraway neighbors always exist in VFA and considered the main reason behind the stacking problem. In, EVFA the virtual forces between the sensor and its distant neighbors are decreasing with the farther distance. Therefore, the stacking problem is effectively solved by EVFA.
In [135], a Distributed Virtual Forces Algorithm (DFVA) is proposed for redeploying sensors after initial random deployment. DFVA aims to enhance the initial coverage and ensures network connectivity. Each sensor node is considered an autonomous entity that participates in constructing the solution by sending and receiving messages. In DFVA, sensors change their positions according to the principle of VFA until they reach a uniform coverage. The authors claimed that if full coverage is achieved and the communication range is twice the sensing range, the network connectivity is ensured. DFVA solves the node oscillations problem by introducing a threshold to limit the distance that a sensor node can travel. This in return, reduces the energy consumption during sensor movements. However, the depleted energy of sensors in DFVA is still higher than that in VFA due to the high mobility during deployment.
Another virtual force-based algorithm using Van der Waals force is proposed in [136] to provide an optimal solution for the deployment problem. The friction force is introduced to the force equation to achieve a steady state of node deployment. Besides, Delaunay triangulation

was adopted to define the relationship of adjacency of nodes because Van der Waal originates only from the interaction between two adjacent nodes. In this algorithm, sensors move to new locations according to the acceleration produced by the repulsion and attraction forces. The proposed algorithm has fast convergence and can achieve a high coverage degree.
Another variant of the VFA named Improved Virtual Force Algorithm (IVFA) is proposed in [137] for maximizing the area coverage. IVFA addresses the shortcomings of the original VFA including, slow convergence, neglecting the boundary effect, useless moves of sensors, and the non-guarantee of virtual force effective distance between sensors. The authors proposed a new exponential force equation to allow the algorithm to converge faster to a steady state. Besides, they introduced a maximum number of steps that a sensor is allowed to perform in each iteration to restrain useless moves. Moreover, IVFA is designed to limit the scope of virtual forces where each sensor node is affected only by the virtual forces of the nodes in its communication range. Finally, the boundary effect is considered by setting up the maximum boundary coordinates.
In [138], a hybrid algorithm was applied for the deployment of large-scale WSNs. The proposed algorithm combines two virtual physical forces namely the dusty plasma crystallization force and the Lennard–Jones Potential force. The deployment strategy takes advantage of the two algorithms in order to organize the sensors in a uniform hexagonal topology. In the hybrid algorithm, the interactions among physical sensor nodes are used to achieve a higher coverage rate and make the equilibrium distance between two nodes more uniform. To achieve these goals, the algorithm starts by deploying the sensors randomly in a circular area, and then, based on the force equations of the two algorithms, sensors are redeployed to form a final optimal hexagonal topology. Comparison between the force-based techniques is illustrated in Table III.3.

Table III.3: A comparison between force-based deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[133]	✓				-Improve the initial coverage -The consideration of obstacles and coverage holes	-High consumption of the sink's energy due to the computation of the new positions -Works with mobile sensors only

[134]	✓	✓			-Enhance network coverage while ensuring connectivity -Solve the node stacking problem -EVFA can be applied with different ratio value of communication range and sensing range	-High computation complexity -Undesirable performance for heterogeneous WSN
[135]	✓	✓			-Achieve good coverage and guarantee network connectivity -It does not require any centralized entity	-More expensive than the centralized VFA
[136]	✓				-Achieve full coverage -Converge faster	-Does not consider obstacles
[137]	✓				-Provide high coverage rate -Fast convergence even in large WSNs	-High computation complexity compared with VFA
[138]	✓				- Achieve higher coverage rate -Provide a uniform deployment for large-scale WSNs	-Does not deal with obstacles -High mobility

III.5.4 Deployment strategies based on integer linear programming

Linear programming (LP) techniques are a class of mathematical techniques used for determining the best solution to a problem through satisfying a series of linear equations and inequalities [139]. A sub-class of LP techniques named integer linear programming (ILP) are used to solve the deployment problems in WSNs where the requirements of coverage, cost, lifetime, and network connectivity must be satisfied. ILP is simply an LP in which the variables take integer values. The deployment problem is solved using ILP through the formulation of the problem and the network settings in a model using a set of constraints and equations. The main characteristic of ILP problems is that their resolution time is very important due to the multiple complicated constraints.
In [103], a deployment strategy based on a mixed-integer linear programming (MILP) model is proposed for target coverage with a minimum number of sensors. Authors assume that the target region is composed of a finite number of objects. Their goal is to cover the objects with the minimum number of sensors while ensuring connectivity between the sensors and the sink. For this purpose, the authors designed two models each of which contains a set of constraints that ensure the minimization of cost, objects coverage, full connectivity, and avoiding model inconsistency. The only difference between the two models is in considering the network lifetime. In the first model, no lifetime constraints are imposed, whereas the second model considers the network lifetime as the period before the first sensor depletes its energy. The presented results show that these techniques achieve good deployment solutions under coverage, connectivity and network lifetime requirements.
An integer linear programming model is developed in [140] for solving the problem of area coverage with a minimum number of sensors. In this model, the deployment area is regarded as a grid that contains a finite number of points. Covering the grid points and minimizing the total number of deployed sensors are taken as the main objectives of this strategy. Furthermore, the authors introduced a set of constraints to guarantee the connectivity between the nodes and to ensure that each point in the grid is covered by at least one sensor. Besides, the authors proposed an improved version of their integer linear model that aims to improve its resolution computation time by reducing the number of decision variables. The authors indicated that the proposed model could be applied to solve k-coverage and m-connectivity problems just by changing a constraint and adding some other constraints. However, an additional number of sensors is needed to achieve k-coverage and m-connectivity.
In [141], an integer linear programming model is proposed to achieve k-barrier coverage in WSNs. The authors attempt to cover a rectangular two-dimensional belt region with minimum relocation of sensors. For this purpose, they used the weighted barrier graph to represent the network where the weight of each edge in the graph is taken as the number of sensors needed to be relocated to connect the two vertices of the corresponding edge. In this algorithm, k-coverage is achieved by the formation of a k sensor-disjoint barrier. Thus, the model objective is to relocate the minimum number of sensors to form the k-coverage barrier. A set of constraints are introduced that every sensor-disjoint barrier should satisfy. The constraints ensure that each sensor should belong to at most one barrier and that the total number of barrier path nodes satisfies the constraint bounds. However, the authors did not consider other important

constraints such as battery constraints and the influence of the surrounding environment. A comparison between the aforementioned integer linear programming techniques is summarized in Table III.4.

Table III.4: A comparison between integer linear programming techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[103]	✓	✓	✓	✓	-Achieve full coverage for targets -Consider connectivity, cost, and network lifetime	-Very high consumption complexity especially when the number of targets is large -Requires long amount of time to find a reasonable solution
[140]	✓	✓	✓		-Solve 1-coverage and k-coverage problems -Ensure network connectivity	-High consumption complexity -Does not consider obstacles -Not efficient when deployment area is large
[141]	✓	✓			-Achieve k-barrier coverage -Ensure network connectivity	-The formulation is computationally intractable due to complicated constraints -High energy consumption

III.5.5 Bio-inspired deployment strategies

The importance of the deployment problem can be easily noticed by observing the growth of the volume of works presented by researchers that attempt to solve this problem. The bulk

of works tackled the problem of deployment through using a special type of optimizers called Bio-inspired optimization algorithms. As we discussed earlier, these optimizers can achieve an excellent approximation of the optimal solution in a reasonable computational time.
Finding approximate solutions to the deployment problem using bio-inspired algorithms is considered a very active search field. The literature provides a huge number of techniques based on bio-inspired algorithms for solving the deployment problem in WSNs. The works presented in the literature offer several deployment solutions that differ according to the optimization objective, the method, and the application where the WSN is implemented.
There are several classifications of bio-inspired deployment techniques. In this chapter, we classify the different bio-inspired deployment techniques based on the method used to obtain the solution. Regarding the method, a big portion of deployment techniques is based on swarm intelligence algorithms. Therefore, in our classification, the techniques are categorized into two main categories: swarm-based deployment techniques (SI) and non-swarm deployment techniques (NON-SI). The latter includes evolutionary algorithms such as Genetic Algorithm (GA) and plants-based algorithms such as Flower Pollination Algorithm (FPA).

III.5.5.1 Swarm intelligence deployment techniques

Despite of their source of inspiration, rules, and concepts, all swarm intelligence algorithms use a population of individuals in order to search for the global optimum in the problem's search space. When solving optimization problems, there are two possibilities to represent solutions. The first one is using all the individuals in the population for representing a potential solution. After the termination of the optimization process, the final positions of all the individuals in the population are returned as a solution to the problem at hand. The second and the most common strategy for solution representation is considering each individual in the swarm as a potential solution to the problem being optimized.
The authors that addressed the problem of deployment used the two representation strategies for representing a potential solution to the deployment problem. The deployment solution refers to a set of coordinates for the sensors in a two-dimensional space.
Suppose we have a set of sensors to be deployed in the area of interest. On the one hand, when using the first representation, each sensor is considered as an individual search agent in the optimization algorithm. This means that the number of individuals in the population exactly equals the number of sensors. In this case, each individual is characterized by two coordinates (x, y), and the final positions of sensors are those of the population's individuals in the search space.
On the other hand, when each search agent in the swarm is taken to represent a potential solution individually, each search agent is considered as a vector that contains the position coordinates of all the deployed sensors. Figure III.8 shows the solution representation used by each search agent.

x_1	y_1	x_2	y_2	x_3	y_3	x_4	y_4

Figure III.8: SI solution representation.

Where x_i and y_i represent the coordinates of the i-th sensor in 2D space. After the representation of solutions, each optimization algorithm employs a set of operators and rules in order to optimize the defined deployment objective (s) and obtain a good approximation of the global optimum for the deployment problem.

Particle Swarm Optimization
The authors in [142] implemented the Particle Swarm Optimization (PSO) algorithm to achieve the maximum coverage using the minimum number of sensors. In this technique, the elements of the particle's position vector are used to represent a solution to the deployment problem. The monitored area is divided into R regions, and each region has to be covered by one sensor. The sensing range of a sensor is defined as the distance between the sensor node, and the farthest location point is his region. The objective is to minimize the sensing range and determine the optimum locations for deploying all R sensors such that every location point is covered.
To avoid premature convergence of PSO and to achieve a better coverage ratio, an improved particle swarm optimization is presented in [143]. The algorithm is based on the ideology of GA. The key of the IPSO is that if the algorithm falls into stagnation after predefined cycles, one particle is chosen randomly, and it will be altered to generate a new particle. If the fitness of the new particle is improved, the variation operation is successful, else, this procedure is repeated until the algorithm achieves a better fitness value.
The authors in [144] presented an improved binary particle swarm optimization for the deployment of stationary sensors in WSNs. The proposed method determines the optimal locations of sensors to meet the desired detection requirements. The proposed method combines the characteristics of the binary particle swarm optimization with the WSN's deployment requirements in order to derive an efficient sensor placement algorithm. The algorithm adopts a new position updating procedure to compute the minimum number of deployed sensors along with their locations. The goal is to improve the detection capabilities and avoid premature convergence.
In [145], a Discrete Particle Swarm Optimization Algorithm (DPSO) for optimal deployment of WSNs in a non-convex region is presented. In DPSO, the position of each particle is limited to a discrete set of values. The area of interest is divided into a finite number of points by a grid of imaginary lines. The grid points denote the potential locations of sensors, and their coordinates are the allowed discrete set of values for the particle position. The proposed method can improve the coverage and reduce the computational complexity since not all the points located inside the area need to be evaluated.
The work in [146] proposed two improved versions of the classical Particle Swarm Optimization algorithm (PSO) called cooperative PSO and cooperative PSO using fuzzy logic to solve the problem of target coverage in WSNs. The optimal locations of sensors that cover the set of targets are determined by optimizing the network lifetime. The two algorithms are designed to

prolong the network lifetime and solve simple coverage, K-Coverage, and Q-Coverage problems.

Artificial Bee Colony Algorithm

Ozturk et al. were the first to use the Artificial Bee Colony metaheuristic (ABC) to tackle the deployment problem in WSNs. In their first study [147], they considered a WSN composed of both stationary and mobile sensors with a probabilistic detection model, which gives a more realistic results. Whilst in their second study [148], a binary detection model was adopted in a network of mobile sensors. In both studies, the position of a food source is considered as a potential solution to the deployment problem, and the ABC metaheuristic is employed to direct sensor movement in order to maximize the coverage area.

The authors in [149] proposed a dynamic deployment strategy for positioning mobile sensors based on a modified ABC algorithm. The goal of the M-ABC is to improve the search capability of the standard ABC algorithm. Two shortcomings of the ABC algorithm are addressed namely: the undirected search for new food sources around old ones and the way the algorithm balances exploration and exploitation. To overcome these shortcomings, a hybrid local search with a crossover operator is presented. Therefore, instead of searching randomly around the old solution, the proposed hybrid local search combines the local search of the standard ABC and the search guided by the global best for balancing exploration and exploitation. Besides, the crossover operator is used to generate new solutions from the fittest ones that have already been found.

The work in [150] presents a parallel and cooperative parallel artificial bee colony algorithm for sensor deployment based on a migration process. The author manages to achieve a better convergence speed by dividing the entire colony into small colonies of equal size and simultaneously evaluating them. In addition, the migration operator is used to substitute the worst solutions in each sub-colony with the best solution discovered in their neighboring sub-colonies. This strategy assists the algorithm in enhancing the solution quality in each sub-colony, thereby, enhancing the quality of the final solution.

The problem of coverage and connectivity of a set of target points is investigated in [151]. The authors highlighted the two shortcomings of the Artificial Bee Colony (ABC) algorithm when using the traditional roulette wheel selection mechanism: the entrapment in local optima and the fast convergence. To overcome the shortcomings, a modified ABC is proposed where the roulette wheel selection mechanism of the follower bees is replaced with the free search algorithm pheromone sensitivity model. The proposed improvement enables the proposed algorithm to achieve higher coverage and connectivity.

The quick Artificial Bee Colony algorithm (qABC), which is a variant of the standard ABC algorithm, was introduced in [152] to solve the problem of deployment in WSNs. The authors considered both the Boolean and the probabilistic detection models. The deployment optimization is taken as a minimization problem where the aim is to minimize the uncovered area, thereby maximizing the coverage of the WSN. The qABC proved that it is effective in optimizing the coverage. However, it needs more CPU to perform optimization.

Ant Colony Optimization Algorithm

In [153], a deployment method based on the Ant Colony Algorithm (ACO) called EasiDesign is proposed. In EasiDesign, the sensing field comprises discrete points that must be covered by sensors. At the beginning of the algorithm, the ants are placed on the sink, and they start building solutions to the problem by moving from one point to another. Each point visited by the ant is a candidate position of a sensor node. Thus, the set of all points visited by the ant is a solution to the deployment problem. The ant moves into locations that enable it to cover the maximum number of points in the detection field. At each step, the ant evaluates the solution using a fitness function. Then, it deposits a quantity of pheromone proportional to the quality of the solution (path). This pheromone information will guide the future ants to move to positions that improve the quality of the solution. Once each ant has built a solution, the solution with the minimum number of sensors is selected. This procedure is repeated until the maximum number of iterations is reached.

The work in [154] also achieved an optimal deployment of WSNs by designing an ant transfer method with three classes of transitions (ACO-TCAT). The idea of ACO-TCAT is the same as EasiDesign, where the ants have to build solutions to the problem by moving in a sensing field that comprises a discrete set of points that must be covered by sensors (Figure III.9a). In addition, instead of the simple selection mechanism of the traditional ACO, the authors presented three novel types of ant's movement. The proposed method aims to decrease inferior solutions and narrow the searching range of the algorithm. As an example, a solution to the problem returned by ACO-TCAT is shown in Figure III.9b.

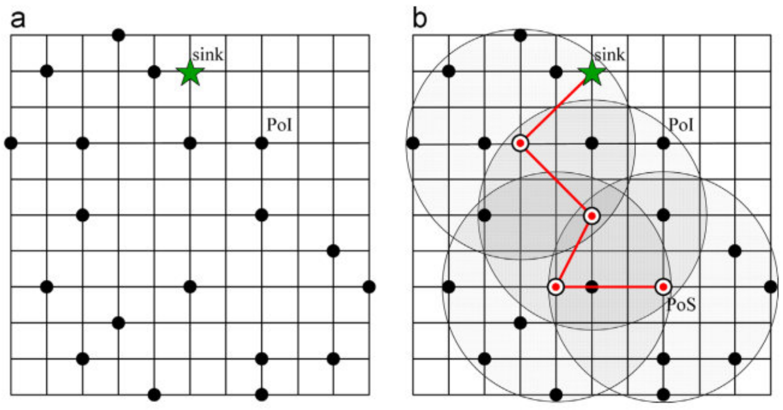


Figure III.9: An example of ant deployment. (a) Before node deployment and (b) After node deployment.

The authors in [155] presented a deterministic deployment method based on the ant colony optimization algorithm. They extended their previous work [154] to consider the problem of grid-based coverage with low-cost and connectivity-guarantee (GCLC). In the proposed method, the region of interest contains a number of grid points that must be covered by sensors. The ants' solutions are constructed incrementally by choosing a set of optimal positions that cover the maximum number of points. For this purpose, in addition to the three types of movement, the authors used a simple greedy migration strategy to cover the set of points with the minimum number of sensors. The novel greedy migration strategy contributes to complete the full coverage quickly and decreases the deployment cost. The influence of the greedy migration strategy on the ant movement is shown in Figure III.10.

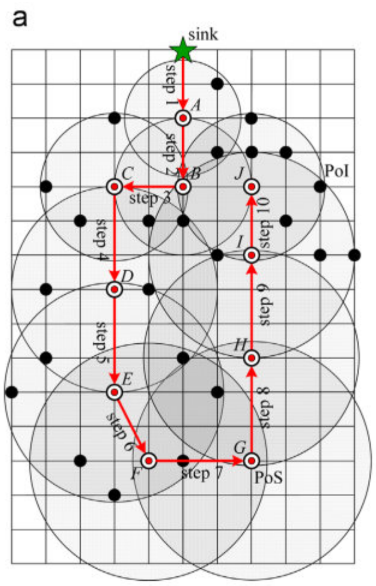


Figure III.10: The influence of the greedy migration strategy on the ant movement.

In [156], a Dynamic Ant Colony Algorithm (DACA) for sensor nodes deployment is proposed. The algorithm includes the integration of new parameters to improve the traditional ant colony algorithm. The integration of the parameters named heuristic factor, expectations heuristic factor, dynamic evaporation factor aims at solving the problem of slow evolution and frequently falling into local optimum.
Aiming at constructing a wireless sensor network that ensures a specified minimum level of reliability during its mission time, the authors in [157] proposed a deployment technique based on the ant colony optimization algorithm. Authors attempt to achieve reliability through minimizing the cost, internal interference, bandwidth usage, and energy consumption during the deployment of the network. As with any ACO-based deployment technique, the ant in this technique starts building a deployment solution from the sink node and concludes the building of the solution when complete coverage of the target points is achieved. After that, the reliability measures are calculated for the constructed solution, and they are compared to the predefined minimum reliability level. If the minimum reliability is achieved, the ant concludes the tours. Otherwise, the ant starts building a new solution. Each ant continues building solutions until the minimum reliability level is met.

Bat Algorithm

In [158], a novel technique based on a smart bat algorithm is proposed for the deployment of WSNs in 3D environments. The authors addressed the problem of deployment by considering several design factors including, network lifetime, coverage, cost, and connectivity. The authors used a set of tabu points to simulate the obstacles in the area of interest. Hence, sensors cannot be deployed at these points. The smart bat algorithm is designed by enhancing the original BA's search mechanism using decision theory and fuzzy logic techniques. The position of each bat is chosen to represent a solution to the deployment problem. In other words, the position vector of each bat contains the coordinates of the grid points where sensors are placed using the smart bat algorithm. The smart bat algorithm is executed with a population of bats generated randomly in the search space with the coverage and the cost as the main objectives. A weight is attached to each objective in order to adjust its importance during the evaluation of the solutions by the objective function.

In [159], a novel deployment technique based on the Bat Algorithm (BA) is proposed. In this technique, each sensor is represented by a bat, and its location is updated based on the steps of the BA metaheuristic. The BA deployment algorithm is designed to maximize the coverage of a set of grid points by minimizing the average distance between the sensors and the grid points. During optimization, the optimal positions of sensors are selected such that the distance between grid points and sensors is small. Besides, the proposed technique reduces the overlapping area between sensors by guaranteeing that each grid point is covered by only one sensor. This is done by ensuring that the grid points covered by one sensor are excluded for the remaining sensor nodes.

Glowworm Swarm Optimization

The work in [160] suggested a deployment technique based on the Glowworm Swarm Optimization (GSO) algorithm. Glowworms in nature carry a luminescent substance called luciferin used to communicate with others. The intensity of the emitted light by a glowworm is proportional to the intensity of luciferin associated with it. Glowworms move according to the luciferin intensity where they are attracted towards the brighter glow of other glowworms in the neighborhood.

Based on this behavior, an efficient deployment technique is proposed where each sensor node is treated as an individual glowworm. In order to maximize the coverage, the authors used the luminescence of the luciferin emitted by glowworms to direct sensors' movement in the area of interest.

After initial random deployment, each sensor receives luminance from its neighboring sensors. The light intensity of the luciferin is related to the overlapping area of the sensor. Sensors with small overlapping areas produce a brighter light than those with the large overlapping area.

In the next step, the sensor selects a neighbor using a probabilistic operator and moves towards the selected sensor. After the movement, the luciferin intensity is updated depending on the overlapping area between the two sensors. In addition, the overlapping area is controlled by reducing the distance between the sensors to be equal to $\sqrt{3r_s}$, where r_s represents the sensing range. The sensors keep moving until coverage above a defined threshold is achieved.

Grey wolf optimizer

An Improved Grey Wolf Optimizer (IGWO) for coverage optimization is presented in [161].

GWO is a swarm intelligence algorithm that mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. In GWO, the position of a wolf represents a candidate solution to the optimization problem. For solving the deployment problem, the position of a grey wolf is represented as a vector that contains the coordinates of the deployed sensors. In order to optimize the deployment efficiently, some improvements are introduced to the original GWO. Firstly, a nonlinear convergence factor is introduced to balance exploration and exploitation. Secondly, the position updating equation of the GWO is improved by an enhanced weighting strategy. Finally, a dynamic variation strategy is used to maintain the diversity of the population and avoids entrapment in local solutions. After applying the IGWO, the position of the grey wolf individual with the largest coverage rate is selected as the final solution.
Another improved grey wolf optimizer is proposed in [162] for deploying WSNs on 3D surfaces. The authors proposed a new deployment technique named the enhanced grey wolf optimizer (EGWO) for optimizing the coverage and the network cost. The EGWO brings several novelties compared with the traditional GWO. The first novelty is the division of the grey wolf population into two parts, one part is responsible for performing the outer-layer encircle, and the second is responsible for the inner-layer encircle. The second novelty is the introduction of chaos to enhance the exploitation and exploration through using Tent mapping. By introducing these improvements, the convergence performance and optimization precision of the algorithm is improved. As an outcome, they led to superior results when solving the problem of deployment in 3D surfaces.

Social Spider Optimization Algorithm

The authors in [163] presented a dynamic deployment technique based on the Social Spider Optimization (SSO) algorithm. The SSO algorithm is inspired by the collective and cooperative behavior of a group of spiders, which interact with each other to perform different activities such as nest building, prey capture, and mating. Each spider is associated with a weight that reflects its quality. Spiders are divided into two categories based on gender: males and females. For solving the deployment problem, the authors represented each social spider individual as a vector that contains all node coordinates in a two-dimensional space. In this technique, the coverage is considered as the weight of social spiders, which refers to the objective in the deployment problem. This means that the higher the coverage, the higher the weight of the spider.
Spiders update their positions based on the communication behavior using vibration. Female spiders present an attraction or dislike based on the weight (fitness) over others irrespective of gender, whereas males are attracted to the closest female spider for mating. Based on this model, sensors change their positions to reduce the blind areas in the sensing region by improving network coverage. The updating process is repeated for several iterations, and the fitter spider is returned as the final solution for the deployment problem.
In [164], a new deployment technique based on the Social Spider Optimization (SSO) algorithm for area coverage maximization is proposed. The proposed technique called real-coded SSO (R-SSO) comes intending to optimize the coverage by changing the nature of the sensing area that is usually modeled as a grid with a discrete nature into a continuous flat surface. This

change assists the algorithm in determining the exact surface of the covered area by reducing the coverage uncertainty of sensor nodes. In the R-SSO, the original steps of the SSO algorithm are implemented to optimize the coverage of an entire area or selected disjointed regions in the area.

Whale Optimization Algorithm
The work in [165] presented a dynamic deployment algorithm based on the Whale Optimization Algorithm (WOA) for optimizing area coverage of homogeneous WSNs. This metaheuristic is inspired by the bubble-net hunting strategy used by humpback whales while hunting. When solving the deployment problem using WOA, the authors supposed that each sensor node is treated as an individual whale. Based on the WOA model, the sensor updates its position according to the positions of other sensors. A fitness interval is used to evaluate the coverage of each sensor. If a sensor node achieves a coverage value in the predefined coverage interval, it is marked as an optimum sensor. Therefore, it will maintain its current location in the following iterations. The sensors will keep changing their positions until no further movement is captured in the network. This deployment strategy is faster than any other technique presented so far. However, it does not maximize the coverage efficiently.

Salp Swarm Algorithm
The Salp Swarm Algorithm is a swarm intelligence algorithm that mimics the swarming behavior of salps when navigating and foraging in oceans. In this algorithm, the position of each salp represents a solution to the problem being optimized. SSA was successfully applied for solving the deployment problem. The problem of deployment is tackled by a variant of the SSA that aims to enhance the search ability and the convergence of the SSA.
A new SSA variant named the Weighted SSA (WSSA) is applied in [166] to tackle the sensor deployment problem in WSNs. In this technique, the coverage and energy efficiency are taken as the optimization objectives. In addition, the location update equation of salps is improved by a new strategy called weighted distance position update. In this improvement, a weight variable similar to the inertia weight of the particle swarm optimization is introduced to balance exploration and exploitation. The WSSA attempts to achieve a trade-off between coverage and energy efficiency in order to achieve better placement of sensors in the sensing area.

Firefly Optimization Algorithm
In [167], a novel deployment technique based on the firefly optimization algorithm is presented to enhance the coverage of sensors that are initially deployed at random in the area of interest. In addition to coverage, the authors considered the minimization of the energy consumed during the movement of sensors as a secondary goal. Therefore, a multi-objective function is designed by combining the two objectives in one function using the weighted sum method.
The working principle of the firefly algorithm is based on the light attraction behavior of fireflies in nature. Each firefly attracts other ones by its brightness, and fireflies with lower brightness move toward the brighter ones. Based on this principle, each firefly is represented by a vector that contains the mobile sensors. The brightness of each firefly reflects the fitness value of the corresponding solution returned by the designed objective function. In this technique, the

positions of fireflies are updated based on the light attraction behavior of fireflies, and the firefly that represents the best individual is returned as a final solution to the deployment problem. A comparison between swarm intelligence techniques is presented in Table III.5.

Table III.5: A comparison between the different swarm intelligence deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[142]	✓				-Enhance the network coverage	-Premature convergence
[143]	✓				-Alleviate the Premature convergence problem	-Not preferable in situations where a full coverage is required
[144]	✓		✓		-Achieve good coverage with least number of sensors	-Does not consider mobility
[145]	✓				-Fast deployment	-Not preferable in situations where a full coverage is required
[146]	✓			✓	-Prolong the network lifetime	-Provide medium coverage quality
[147]	✓				-Give more realistic results	-Not preferable in situations where a full coverage is required
[148]	✓				-Enhance the network coverage	-Require a lot of iterations
[149]	✓				-Enhance search capability of the standard ABC	-Undesirable performance for large WSNs

[150]	✓				-Achieve higher coverage	-High computation complexity
[151]	✓	✓			-Enhance network coverage while ensuring connectivity	-High complexity due to complicated constraints
[152]	✓				-Solve the problem of deployment effectively	-Needs more CPU to accomplish the optimization task
[153]	✓				-Achieve good coverage	-Increase the number of deployed sensors
[154]	✓	✓			-Enhance the network coverage	-Require more steps compared with ACO
[155]	✓	✓			-Reduce the number of deployed sensors	-Increase the overlapping area
[156]	✓				-Solve slow evolution problem of the ACO	-Not preferable in situations where a full coverage is required
[157]	✓			✓	-Achieve high level of reliability	-Require a lot of evaluations compared to the mentioned ACO-based techniques
[158]	✓		✓	✓	-Solve the problem of deployment in 3D surfaces	-Not preferable in situations where a full coverage is required

[159]	✓				-Fast deployment	-Provide medium coverage quality
[160]	✓				-Achieve full coverage	-High mobility
[161]	✓				-Optimize the coverage effectively	-Suffer from slow evolution
[162]	✓		✓		-Solve the problem of deployment in 3D surfaces	-Works only in simple 3D surfaces
[163]	✓				-Enhance the network coverage	-Premature convergence
[164]	✓				-Determine the exact network coverage	-Not preferable in situations where a full coverage is required
[165]	✓				-Fast deployment	-Provide medium coverage quality
[166]	✓			✓	-Provide good coverage and minimize energy consumption	-Undesirable performance for large WSNs
[167]	✓			✓	-Enhance the network coverage	-Fail to achieve a good trade-off between the objectives

III.5.5.2 Non-Swarm intelligence deployment techniques

This category of deployment methods contains several types of algorithms arranging from the popular GA to the plants-based algorithms such as Flower Pollination Algorithm (FPA). Different works have been proposed based on non-swarm intelligence algorithms to tackle the problem of deployment in WSNs. In the following paragraphs, we will review these works and we will explain how each one of them has addressed the deployment problem.

Genetic Algorithm

The authors in [168] presented an efficient technique based on a genetic algorithm for solving the coverage holes problem in WSNs. the authors addressed the problem of coverage holes caused by the random deployment of static sensors. The idea of their proposed algorithm is to determine the minimum number of mobile sensors that must be added after the initial deployment and their optimal locations using the genetic algorithm. Each chromosome, as a solution in the GA, represents the location of a potential mobile sensor in the sensing field. The values of the location's coordinates are represented using binary digits. The two well-known GA operators are implemented in this technique. The crossover operator is used to create new individuals, and the mutation operator is applied to avoid trapping in the local optimum.
In [140], a deployment technique based on a Genetic Algorithm (GA) for solving the grid coverage problem with the least number of deployed sensors is proposed. In this proposition, each chromosome, as a solution in the GA, represents a solution to the deployment problem. Hence, each chromosome is regarded as an array that contains all the sensors' coordinates. The steps of the classical GA are implemented with a new chromosome generation algorithm for minimizing the number of deployed sensors. The GA-based deployment technique deterministically deploys the sensors to achieve the maximum possible coverage.
The work in [169] tackled the deployment problem using Non-Dominating Sorting Genetic Algorithms (NSGA-II) in an indoor environment with obstacles. In NSGA-II, the authors dealt with the deployment problem by optimizing the coverage, cost, and connectivity. To take advantage of the GA operators, each chromosome, which encodes the positions of mobile sensors, is represented by a binary string. Additionally, a multi-objective function is designed to combine the three objectives in order to achieve an optimal deployment with maximum coverage using fewer sensor nodes.
The authors in [170] proposed a novel genetic algorithm for area coverage maximization in heterogeneous WSNs. They employed the GA with multiple crossover operators and a new heuristic initialization of individuals to maximize the area coverage. In addition, they introduce a new fitness function to calculate the exact area covered by the set of sensors. Besides, the VFA is applied as a local search algorithm to improve the final solution quality.
In [171], an improved genetic algorithm for area coverage maximization with a minimum number of deployed sensors is proposed. A variable-length encoding is used in this technique to represent the size of chromosomes. Each chromosome as a solution in the genetic algorithm is utilized to represent a deployment sequence of mobile sensors. For solving the deployment problem, the authors focused on maximizing the coverage, minimizing the overlapping area, and reducing the number of sensors. To achieve the above-mentioned goals, a two-point crossover operator is proposed to produce new individuals with a variable number of sensors. Besides, the connectivity is ensured by employing a penalty to the objective function.

Bacterial Foraging Optimization

In [172], a deployment technique for finding optimal locations that reduce the sensing range of sensors is proposed. This technique optimizes the deployment using the Bacterial Foraging Optimization (BFO) algorithm. The BFO algorithm is inspired by the foraging behavior of

Escherichia Coli bacteria that live inside the human intestine. In this technique, the bacteria is used to find the optimal coordinates of sensors inside the area of interest. It is supposed that the area contains a finite set of location points where the sensors can be deployed. The objective is to determine an optimal subset from the set of location points that enhance the coverage and reduce the sensing range of sensors. Initially, a population of bacteria is initialized randomly in the search space. Then, the bacteria start to swim to the next position if the next position is best otherwise, it tumbles to the other direction. The swimming and tumbling are repeated until all the bacteria converge towards a single point in the area. This point is an optimal position where a sensor is to be deployed. These steps are repeated until all the sensors are deployed.

The work in [173] proposed another deployment technique based on the bacterial foraging optimization algorithm for maximizing area coverage. In the proposed technique, the sensors are considered as the bacteria that are in search of food. The area of interest is partitioned into regular hexagons, and the sensors are supposed to occupy the vertices of the hexagons in order to ensure communication and complete coverage. The sensors are initially deployed at random, and their positions are updated according to the steps of the BFO algorithm. Once a sensor reaches the vertex of a hexagon, it will stop moving and this sensor is moved into a queue. If two sensors (bacteria) reach the same vertex, then, they produce a new sensor with double battery power but a single communication link as per BFO. When all the sensors are in the queue, it means that all the sensors have reached some vertex. Therefore, the BFO deployment algorithm converges, and the optimal positions of sensors are returned.

Biogeography-Based Optimization Algorithm

In [174], a Biogeography Based Optimization (BBO) algorithm is applied to the dynamic deployment of WSNs including both mobile and static sensors. In the standard BBO algorithm, each candidate solution to the optimization problem is called a "habitat" (or island). The goodness of each solution is called the habitat suitability index (HSI), which corresponds to the fitness value of the solution. The habitat with a high HSI is considered optimal.

In the BBO-based placement algorithm, the deployment of sensors in the area of interest refers to a habitat (a solution in the BBO algorithm), and the coverage rate of the network corresponds to the fitness value (HSI) of that solution. The algorithm starts by deploying a number of stationary and mobile sensors randomly in the sensing area. Then, the migration and mutation operators of the BBO are used to move the mobile sensors to optimal locations that optimize the network coverage. Migration operator is used to modify existing solutions using both immigration and emigration rates, whilst mutation is a probabilistic operator that randomly modifies current solutions to produce new ones.

The work in [175] proposed a new technique for deploying WSNs using biogeography-based optimization (BBO) algorithm. The goal of the proposed technique is to solve the connected coverage problem by ensuring k-coverage and m-connectivity requirements. In this technique, the area of interest contains a set of target points that must be covered by sensors, and the objective is to organize the sensors by finding an optimal number of suitable positions such that the network requirements are satisfied. To achieve this purpose, the authors designed an objective function with some constraints in order to guarantee the coverage and connectivity requirements. Besides, the steps of the BBO are implemented with an efficient encoding scheme for representing the habitats, and then the BBO's migration and mutation operators are used to optimize the defined objective function.

Immune Algorithm

In [176], a deployment technique that aims to maximize the coverage of a set of sensors placed randomly in the area of interest is proposed. In this technique, the Immune Algorithm (IA) is used to move the mobile sensors into optimal positions after initial random deployment. The proposed technique determines a set of optimal positions that minimize the amount of energy consumed by the sensors to reach them. For this purpose, each antibody as a solution in the immune algorithm is encoded as a vector of floating-point numbers representing the position coordinates of the sensors in the area of interest. The popular operators of the immune algorithm namely: selection, replication, mutation, and elitism, are repeated until the coverage value doesn't change for a certain number of iterations or when the number of iterations exceeds the specified maximum iterations.

Flower Pollination Algorithm

In [177], the Flower Pollination Algorithm (FPA) is employed to find the best placement patterns for sensors that maximize the overall area coverage. The ultimate goal of this technique is to achieve the optimal placement of sensors without affecting network connectivity constraints. The authors used a deterministic placement strategy as the first step to guarantee connectivity before the flower pollination algorithm optimizes the deployment. In order to perform coverage optimization, authors represent each flower as a solution by a vector that contains the coordinates of all sensor nodes in the area of interest. As a final step, the FPA is applied to optimize the coverage without losing connectivity until a termination criterion is met.

The work in [178] presents two flower pollination algorithms for heterogeneous WSNs deployment with the presence of obstacles. The first proposition is a single-objective improved flower pollination algorithm that aims at maximizing the network coverage. The improved algorithm introduces a chaotic map to maintain the diversity of the population, and a nonlinear convergence factor to deal with the slow convergence of the FPA. In addition, a greedy crossover strategy is adopted to improve the individuals' quality where it replaces some variables with others obtained from superior individuals. The second proposition is a multi-objective algorithm based on non-dominated sorting designed to tackle the problem of deployment in a forest environment. The algorithm aims at maximizing the coverage, minimizing radiation overflow rate, and minimizing the energy consumption of the sensor nodes while maintaining connectivity. Table III.6 summarizes the comparison between the previously mentioned non-swarm deployment techniques.

Table III.6: A comparison between the different non-swarm deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[168]	✓				-Reduce coverage holes	-High computation complexity due to the binary representation
[140]	✓		✓		-Achieve higher coverage with minimum cost	-Not suitable for random deployment
[169]	✓	✓	✓		-Optimize several deployment objectives	-Bad performance in large WSNs
[170]	✓				-Perform well in terms of coverage	-High mobility
[171]	✓		✓		-Achieve good coverage with least number of sensors	-Premature convergence
[172]	✓				-Cover the set of targets effectively	-Designed only for small WSNs
[173]	✓				-Achieve high coverage	-Designed only for areas with hexagonal shape
[174]	✓				-Outputs good coverage results	-Falls easily in local optima
[175]	✓	✓			-Ensure k-coverage and m-connectivity	-High computation complexity
[176]	✓				-Provide acceptable coverage quality	-Increase the overlapping area

[177]	✓	✓			-Enhance network coverage while ensuring connectivity	-Not suitable for random deployment
[178]	✓	✓			-Optimize the deployment effectively	-Designed only for small WSNs

III.5.5.3 Hybrid deployment techniques

In the last few years, researchers have turned their attention toward a new research trend known as hybridization. Hybridization is a ubiquitous concept, where it allows combining the strengths of original methods to construct an efficient solution. Using hybridization, the weakness of an algorithm is compensated by another algorithm and vice versa. Some researchers that addressed the problem of deployment in WSNs have used hybridization between several methods to develop robust and efficient solutions. In the following paragraph, the most efficient and recent hybrid deployment techniques are briefly reviewed.
In [179], a hybrid algorithm for solving the coverage problem in WSNs is presented. The proposed algorithm combines the strengths of the GA and binary ant colony optimization. The algorithm aims to determine the optimal set of active sensors and their locations while ensuring some defined constraints related to the coverage and sensor count. In this hybridization, the operators of the GA namely crossover and mutation, are introduced to the binary ant colony optimization. The main algorithm that constructs the solutions is the binary ant colony optimization, however, the operators of the GA are injected into the ACO in order to enhance the global property of solutions. The proposed algorithm shows its efficiency in enhancing the accuracy of solutions and improving the coverage using the minimum number of working set nodes.
In [180], the problem of 3D indoor deployment of sensors is solved using a well-designed hybrid algorithm. The authors highlighted the shortcomings of the NSGA-III algorithm, which is a variant of the GA when solving mono-objective and two-objective optimization problems, and those of ACO which concerns premature convergence. To alleviate the shortcomings, the authors suggested the hybridization of the two algorithms to take advantage of their strengths in order to construct an efficient solution. In this hybridization, the ACO is employed as an operator in the NSGA-III algorithm. The solution to the deployment problem is constructed incrementally by the ants. The proposed algorithm performs an efficient mutation strategy to speed up the local search and allows finding a suitable solution to the 3D deployment problem.
The work in [181] proposes a Maximal Coverage Hybrid Search Algorithm (MCHSA) for maximizing area coverage using multiple types of sensors. The MCHSA is constructed based on the framework of the Particle Swarm Optimization (PSO) algorithm and the Hooke Jeeves method. The latter is applied to overcome the stagnation problem of the PSO algorithm. That is, if the PSO search stagnates, the Hooke Jeeves method is called to improve the fitness of the global best. MCHSA has the advantage of maximizing the coverage using multiple types of sensors.

However, it takes a longer computational time because of the repetitive calling of the Hooke Jeeves method.
The authors in [182] employed two nature-inspired algorithms to optimize the area coverage in heterogeneous WSNs. The first algorithm is an Improved Cuckoo Search (ICS) derived from the original CS algorithm in which some important parameters are functionalized to enhance the convergence rate. The second is a Chaotic Flower Pollination algorithm (CFPA), where the global searching mechanism of the original FPA is improved. In this improvement, the global pollination equation is altered by considering the information of the whole population instead of only the information about the best individual. After optimizing the coverage using both algorithms, the authors applied the Virtual Force Algorithm (VFA) to reduce the overlapping area between sensors.
In [183], a deployment technique is proposed by hybridizing the classical GA and the binary particle swarm optimization algorithm. The proposed technique labeled GA-BPSO, is designed to find the optimal locations of sensors while maximizing both coverage and connectivity. The solution in GA-BPSO is considered as a sequence of potential positions, where the sensors can be deployed and it is encoded as a binary vector. When the i-th vector cell has the value 1, it means that the i-th potential position is selected to deploy a sensor. Consequently, the vector cell with the value 0 means that the corresponding potential position is not selected to deploy a sensor. The GA-BPSO works as follows: a population of solutions is generated, then, the population is divided into two parts of equal size. One sub-population contains the best individuals and the other one contains the rest of the population. The sub-population with the best individuals is used as input for the GA, while the sub-population with the worst individuals is used as input for the BPSO algorithm. After the run of both algorithms, the outputted sub-populations, which contain the best solutions returned by the two algorithms, are merged into one population and serves as the input population for the GA-BPSO. A detailed comparison between hybrid deployment techniques is provided in Table III.7.

Table III.7: A comparison between hybrid deployment techniques.

Technique	Objectives				Advantages	Disadvantages
	Coverage	Connectivity	Cost	Lifetime		
[179]	✓			✓	-Achieve good coverage and enhance the accuracy of solutions	-Long computational time due to the binary representation
[180]	✓				-Enhance the coverage in 3D surfaces	-Not suitable for dense deployment

[181]	✓				-Maximize the coverage of heterogeneous WSNs	-Long computational time due of the repetitive calling of the Hooke Jeeves method
[182]	✓				-Achieve high coverage	-High mobility
[183]	✓	✓			-Solve k-coverage and m-connectivity problems	-Undesirable performance for large WSNs

III.6 Conclusion

In this chapter, we have reviewed in detail the problem of deployment in wireless sensor networks. We have also presented an up-to-date review that summarizes the current status of the research works presented in the literature to resolve this issue. Furthermore, we have highlighted the strengths and weaknesses of the different deployment techniques by putting in place a set of clear and useful comparisons.

The optimal deployment of WSNs was defined as an NP-hard optimization problem in most works in the literature. As can be seen from the presented state of the art, both exact and bio-inspired techniques have been used by researchers for obtaining the optimal solutions for the deployment problem. Although the problem of deployment is intensively studied, no method is designed for solving all types of deployment problems. Therefore, the issue of deploying WSNs remains largely open to possible improvements where more effort is required to achieve optimal deployment patterns. From a theoretical point of view, the possible improvements concern either the quality of the solutions or the effectiveness of the resolution technique. Those two improvement possibilities are the subject of our contributions that are presented in the following chapters for solving the deployment problem.

CHAPTER IV

OPTIMAL DEPLOYMENT OF HOMOGENEOUS AND HETEROGENEOUS WIRELESS SENSOR NETWORKS

Contents

IV.1 Introduction

Effective deployment of Sensor Nodes (SNs) is a major point of concern, as the performance of any WSN primarily depends on it. This chapter is devoted to solving the deployment problem in WSNs. We address the deployment problem by maximizing the area coverage in both homogeneous and heterogeneous mobile WSNs. We present a novel technique based on an Improved Bees Algorithm (IBA) for providing optimal deployment patterns regardless of the number of deployed sensors and their initial locations.

In order to validate the proposed technique, it is compared with recently proposed deployment techniques through several experiences in terms of coverage, convergence, and stability.

IV.2 Problem description

The main objective of any wireless sensor network is to ensure continuous monitoring of an area or for a given set of targets. Coverage is considered an important Quality of Service (QoS) criterion for many WSN's applications. To ensure that the required QoS is achieved, the sensors must be placed in locations that optimize the area coverage and eliminate the coverage holes, especially when deploying sensors with heterogeneous sensing ranges.

As we explained in chapter III, WSNs are classified into two categories: homogeneous and heterogeneous WSNs. The heterogeneous sensor network is more demanded in real-life applications compared with a WSN with fixed sensing capabilities. Moreover, the use of heterogeneous sensors is more practical and provides more flexibility to the network. For example, rather than using expensive sensors with high characteristics, we can construct a mixed network with different kinds of sensors to achieve a balance between performance and cost.

Regarding deployment, the deployment of heterogeneous sensors is much difficult compared with the deployment of homogeneous sensors. The problem of redundant coverage (sensors covering the same targets) usually occurs when deploying a set of heterogeneous sensors (sensors with multiple sensing ranges). This problem becomes a main issue that has a great influence over the detection performance and the deployment cost. In this problem, a sensor with a small sensing range might be placed in a location where its sensing zone overlaps partially or completely with the sensing zone of another sensor with a bigger sensing range (Figure IV.1). The deployment strategy has to ensure the choice of the appropriate positions for the sensors despite their sensing ranges to maximize the covered area. Therefore, our interest is focused in particular on the problem of maximizing the area coverage in homogeneous and heterogeneous WSNs.

In this study, we investigate the deployment problem in WSNs by presenting a novel sensor deployment scheme based on an Improved Bees Algorithm (IBA) for optimizing the area coverage after an initial random scattering in both homogeneous and heterogeneous WSNs. The IBA metaheuristic includes a neighborhood shrinking procedure that aims to improve the local search efficiency and a site abandonment procedure for enhancing the ability of the algorithm in avoiding the entrapment in the local optimum.

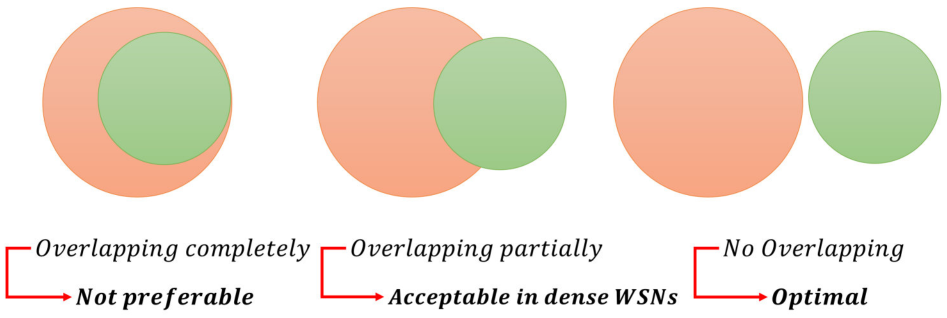


Figure IV.1: Overllaping vs non-Overllaping Coverage

IV.3 Proposed Sensor Deployment Technique

In this section, IBA-based technique for solving the problem of deployment in WSNs is discussed. First, we present the network assumptions and the coverage model. Then, the steps of the proposed deployment scheme are described in details.

IV.3.1 Network assumptions

The initial network assumptions of the proposed model are:

- The network is composed either of a number of homogeneous sensor nodes with the same sensing capabilities or a number of heterogeneous sensors with different sensing capabilities.
- All sensor nodes are mobile and have the ability to change their positions during deployment.
- The sink node is placed in a predetermined position in the sensing region.
- Initially, the sensors are scattered at random over the sensing area.

IV.3.2 Coverage model

For coverage calculation, we utilize the Boolean perception model to define the detection capability of a sensor. The Boolean perception model in WSNs describes the probability of detecting targets by a sensor node. In this model, a perception circle with the sensing range as its radius is drawn around the location of each sensor. As illustrated in Figure IV.3, the sensor can only detect events within its perception circle (white shape). Consequently, the other events that happen outside the circle are undetectable (black shape).

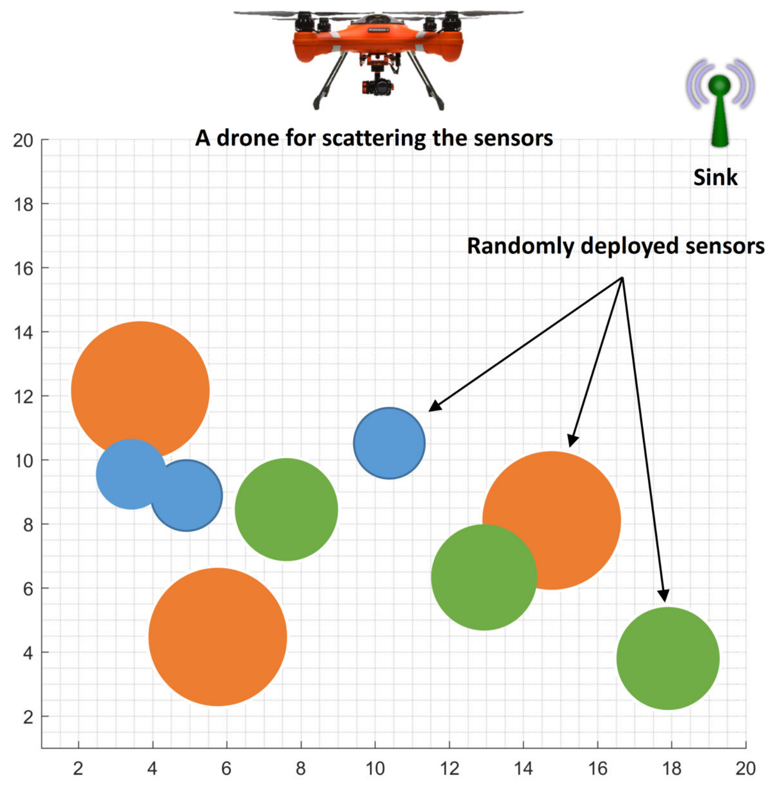


Figure IV.2: Network assumptions.

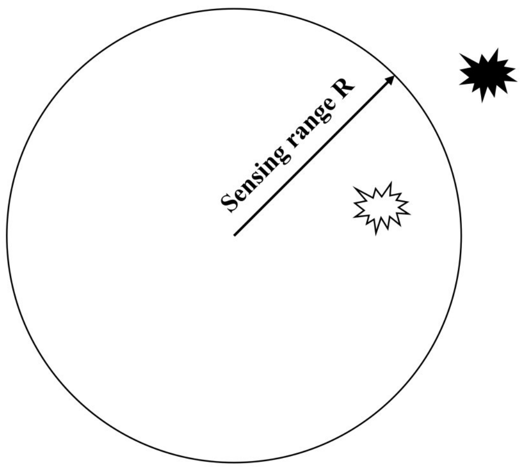


Figure IV.3: Binary detection model.

We assume that the sensing area is a two-dimensional $M \times N$ grid where the distance separating two adjacent points of the grid is equal to 1 unit. Figure IV.4 shows the shape of the area of interest used in our work.

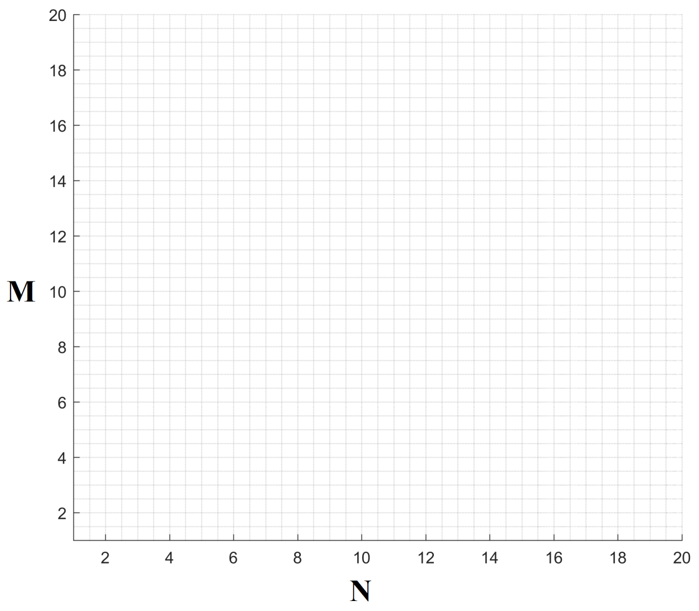


Figure IV.4: The deployment area.

To optimize the coverage, the sensors are supposed to cover the maximum number of points in the grid. The coverage of each point in the grid is judged based on the Euclidean distance separating it from the set of sensors. Considering a target point T located at (x_t, y_t) in the 2D space, the coverage of the point T by a sensor s_i in the Boolean perception model is decided based on the distance separating them. For covering the point T, the distance between the sensor s_i located at (x_i, y_i) and the target point T should be less than the sensing range R_s of the sensor s_i.

$$P(x_i, y_i, x_t, y_t) = \begin{cases} 1, & if\ \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2} < R_s \\ 0, & otherwise \end{cases} \tag{IV.1}$$

To cover the entire area of interest, the deployed sensor nodes have to ensure full coverage of all the grid points. Therefore, given a set of sensors $S = \{s_1, s_2, \ldots, s_{ns}\}$, the probability that a target point T located at (x_t, y_t) is covered by the set of sensor nodes S can be written as:

$$PC(S, x_t, y_t) = 1 - \prod_{i=1}^{ns}(1 - P(x_i, y_i, x_t, y_t)) \tag{IV.2}$$

The coverage is then expressed by the total grid points covered by the set of sensors. Therefore, the coverage rate of the entire area is defined as follows:

$$CR = \frac{\sum_{x_t=1}^{M} \sum_{y_t=1}^{N} PC(S, x_t, y_t)}{M \times N} \tag{IV.3}$$

IV.4 The Bees Algorithm (BA)

The Bees Algorithm (BA) was proposed by Pham and Castellani in 2006. It mimics the behavior of a swarm of honeybees when searching for food sources. In BA, the bees are classified into two categories; scout bees and foragers. The scouts are responsible for discovering new food sources to collect nectar. They start exploring randomly the surroundings of the hive in search for promising flower patches. When a scout finds a good food source, it remembers the position information of the discovered food source and returns to the hive. The scout shares the information about its findings with the foragers that are waiting in the hive. The scout utilizes a special dance called the waggle dance to inform the foragers about the discovered food source. The highly-rated patches that contain rich and easily available nectar (with higher fitness) attract the largest number of foragers. Finally, the scout bee goes back to the flower patch followed by the recruited foragers to collect the nectar. When a forager bee comes back to the hive, it may in turn waggle dance to direct other bees to the flower patch.

The artificial BA starts with ns scout bees being placed randomly in the search space. Then, the position of each scout is evaluated using a fitness function. At each iteration, the nb scouts that discovered the solutions with highest fitness perform the waggle dance to recruit a number of foragers. The ne top-rated solutions recruit the largest number of foragers (nre) to the discovered flower patch. The remaining scout bees recruit ($nrb < nre$) foragers. This mechanism allows a larger number of bees to exploit the most promising areas in the search space.

The foragers are placed randomly on solutions in the discovered flower patch. The flower patch represents the visited food source and its vicinity expressed by the neighborhood size (ngh). If within the neighborhood a forager lands on a solution better than the solution advertised by the scout, this forager becomes the new scout and replaces the old scout in performing the waggle dance in the next iteration. The remaining scout bees (ns - nb) that have found poor food sources are assigned randomly in the solution space scouting for new promising flower patches. The above steps are repeated until a satisfactory solution is found or a maximum number of iterations is reached. The flowchart of the BA is presented in figure IV.5.

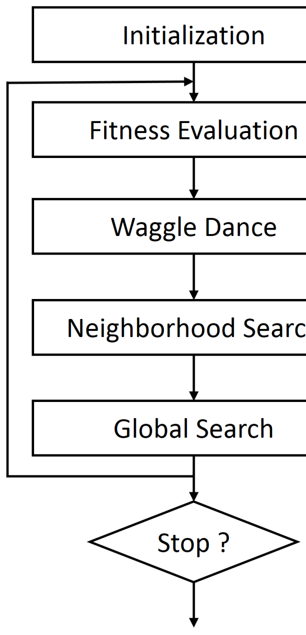


Figure IV.5: Flowchart of Bees Algorithm.

The BA has been widely used for solving different classes of problems, it has been successfully applied to project scheduling problem, multiple disc clutch problem, printed circuit board assembly minimization problem, etc. This makes it suitable for solving different engineering design problems.
The algorithm steps shown in Figure IV.5 are detailed below to solve the sensor deployment problem.

IV.4.1 Solution Representation

In the BA, the position of each food source represents a feasible solution for the problem being optimized. Assume we have $S = \{s_1, s_2, \ldots, s_i, s_n\}$ wireless sensors to be deployed in the area of interest. Each solution $X_i (i = 1, \ldots, m)$ is a vector $X_i = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$ that contains the position coordinates of all the sensors. Where x_i and y_i represent the coordinates of the ith sensor node in 2D space. Figure IV.6 shows the solution representation of five sensors for both homogeneous and heterogeneous cases.

IV.4.2 Initialization

The BA starts by employing a fixed number of scout bees randomly in the search space looking for food sources. As mentioned before, the position of each food source can be identified as a vector of coordinates. Furthermore, in our work we assume that the sensors are deployed randomly over the sensing area. Therefore, given a deployment area of size $M \times N$, the initial coordinates are chosen randomly inside the bounds of the deployment area such that $0 \leq x \leq M$ and $0 \leq y \leq N$.

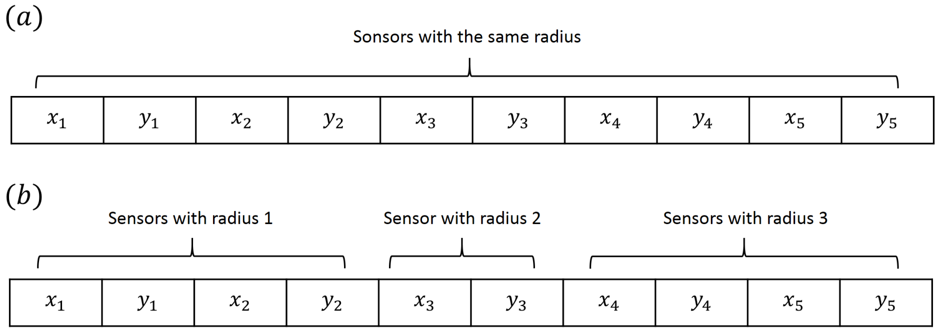

(a)
Sensors with the same radius
x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5
(b)
Sensors with radius 1
Sensor with radius 2
Sensors with radius 3
x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5

Figure IV.6: Solution Representation of IBA: (a) homogeneous case, (b) heterogeneous case.

IV.4.3 Deployment optimization

Similar to other metaheuristics, the search process in BA is divided into local search phase and global search phase. Local search is based on a random distribution of foragers in a pre-defined range. Thus, a flower patch (neighborhood) of size (ngh) is created around the most promising solutions. The flower patch represents the visited food source and its vicinity expressed by the neighborhood size. Then, at each iteration, the scouts that located the solutions of highest fitness perform the waggle dance and recruit a number of foragers to further explore the selected flower patches. This mechanism allows a larger number of bees to search the neighborhood of the most promising solutions.

The recruited foragers are placed on solutions in the vicinity of the solution advertised by the scout. They conduct a local search in the neighborhood in the hope of finding a better solution than the scout.

For creating new deployment solutions in the vicinity of the solution advertised by the scout, the foragers are placed on solutions that are generated by changing the position of one of the sensors in the scout's solution as follows:

$$V_{ij} = X_{ij} + \omega \tag{IV.4}$$

Where X_i is the selected solution, V_i is the new produced solution, ω is a random number in the range $[-ngh, +ngh]$, and j is the randomly selected mobile sensor's position.

By choosing the value of ω in the range $[-ngh, +ngh]$, we guarantee that the generated solutions are located within in the bounds of selected flower patch (Within the neighborhood).

After creating new solutions for the foragers, the quality of the generated solutions are evaluated in terms of coverage. If one of the recruited bees finds a better network coverage than the scout, that recruited bee becomes the new scout and participates in the waggle dance in the next generation. While foragers are recruited to conduct local search, the bees that have found a poor food sources under certain threshold (worst coverage) are assigned randomly in the solution space scouting for new promising flower patches.
The steps used in deployment optimization are illustrated in Figure IV.7.

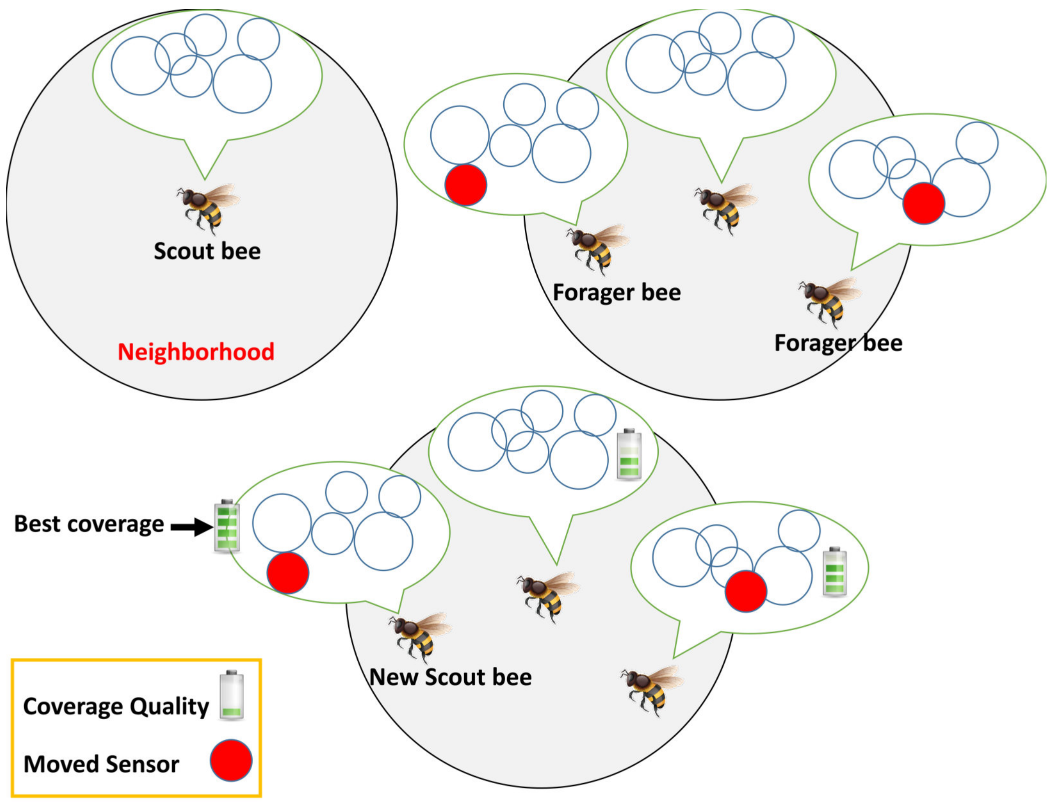


Figure IV.7: Deployment optimization using BA.

IV.4.4 Bee colony population

At the end of each iteration, the selected bee (best solution) from each patch combined with the remaining scout bees assigned randomly to conduct the search will form the new population in the next iteration.

IV.4.5 Stopping criterion

The above steps are repeated until a coverage rate value above a predefined threshold is met or a maximum number of iterations is reached.

IV.5 The Improved Bees Algorithm (IBA)

With the algorithm proposed so far, the locations of sensors are changed according to the steps of the basic BA algorithm. However, these steps are not enough to optimize the coverage effectively due to several reasons. The first is when the chosen neighborhood size is very large. In this case the search space for the forager bees becomes large too, which makes the search inefficient and the local search becomes more like a random search. In this case, the BA loses its exploitative ability, which in turn leads to an unsatisfactory outcome when optimizing the coverage. The second drawback of the BA emerges when the foragers fails to improve the coverage in a particular site. In other words, this drawback emerges when the foragers keeps falling in local solutions. In this situation, there is no feature in BA that allows the algorithm from escaping from these local solutions. This will again lead to an unsatisfactory outcome when optimizing the WSN's coverage.

To overcome these drawbacks and achieve the desired search accuracy, the local search must be concentrated around the best solutions by narrowing the local search space. This mechanism increases the chances of obtaining a good approximation of the global optimum. Furthermore, avoiding the stagnation in local solutions is a major requirement especially when the local search in a flower patch fails to enhance the quality of the produced solutions. Generally, metaheuristic algorithms utilize random changes in the solution to escape from the local optimum.
In this regard, two procedures named neighborhood shrinking and site abandonment are introduced to the basic BA. The two procedures are called when the local search fails to improve the coverage value in a particular flower patch.

IV.5.1 Neighborhood shrinking procedure

The shrinking strategy is proposed to improve the solution quality. Initially the patch size is assigned a large value using the following equation:

$$ngh(0) = (max - min) \tag{IV.5}$$

Where min and max denote the lower bound and the upper bound of sensors coordinates respectively.

During optimization, the initial size is kept unchanged as long as the local search in the neighborhood can improve the solution quality. If the local search does not yield any progress in coverage after a predefined number of cycles, the patch size is shrunk according to the following heuristic formula:

$$ngh(t+1) = 0.8 \times ngh(t) \tag{IV.6}$$

The shrinking procedure promotes exploitation over exploration by narrowing local search space and searching more densely the area around the best solutions.

IV.5.2 Site abandonment procedure

If the neighborhood shrinking procedure fails to improve the quality of solutions and the local search in a particular patch keeps falling into stagnation after a predefined number of iterations, it is assumed that the fitness value of that particular patch has reached a peak. Consequently, that solution is replaced with a new randomly produced one. This procedure prevents the search from being trapped in sub-optimal solutions. If the abandoned solution corresponds to the best global solution it will be stored so that if the algorithm fails to achieve any improvement the stored solution is taken as the final one.

The pseudo-code of the IBA-based deployment algorithm is given in Figure IV.8.

```
Pseudo-code of the IBA

Input: size of area of interest, the number of sensor types, the sensing radius of each
sensor type, the number of sensors for each type, the number of scout bees,
neighborhood size, stagnation limit, and maximum number of iterations MaxIter.

Output: an array contains the optimal positions of the mobile sensors.

Initialize population by deploying S sensors randomly for each food source Xi.
For t = 1 to MaxIter do

1. Evaluate the fitness of the population.
2. Sort the solutions based on their fitness.
3. Select nb solutions with the highest fitness for neighborhood search.
4. Recruit nre foragers for each of the ne elite sites selected among the nb best sites.
5. Recruit nrb foragers for each of the remaining best sites.
6. Produce new solutions for the foragers in the neighborhood of the selected sites
using Eq.(IV.4)
7. Check the feasibility of produced solutions.
8. Select the fittest solution (bee) from each patch.
9. Perform the shrinking procedure using Eq.(IV.6) when the neighborhood search of
the retained solutions does not yield any progress.
10. Abandon sites that keeps falling into stagnation after a limit number of iterations.
11. Record the abandon solution if it corresponds to the best global solution.
12. Allocate the rest of the bees (non-selected sites) randomly for global search,
13. Form the new population.
14. Memorize the best solution achieved thus far.
End.
```

Figure IV.8: IBA-based deployment algorithm.

IV.6 Experimental Results and Analysis for IBA

To observe the performance of the proposed technique in solving the deployment problem, we provide a set of simulations for two types of wireless sensor networks: the first network is a homogeneous WSN in which all sensors have the same characteristics and the same detection capability. Whereas the second network is a heterogeneous WSN with three types of sensors differ in the sensing range. Five related work algorithms are used to assess the performance of the IBA in optimizing the network coverage. Table IV.1 illustrates the information about the comparison algorithms used in the set of experiments. The parameter settings of the algorithms are the same as those in the corresponding publications. In all the experiments, the sensors are deployed in an area with a size of 100m by 100m. The parameter settings of the IBA used for comparisons with related works are presented in Table IV.2.

Table IV.1: The comparison algorithms.

Algorithm	Reference	Sensor type	Coverage type
BA	[184]	homogeneous	Area coverage
qABC	[152]	homogeneous	Area coverage
SSO	[163]	homogeneous	Area coverage
MCHSA	[181]	heterogeneous	Area coverage
IFPA	[178]	heterogeneous	Area coverage

Table IV.2: Parameter settings of the IBA.

Parameter	Value
Scout bees (ns)	7
Best sites (nb)	5
Elite sites (ne)	3
Recruited bees of elite (nre)	7
Recruited bees of best (nrb)	1
Neighbourhood size (ngh)	40
Stagnation limit	10
Shrinking factor	0.8

IV.6.1 Deployment of Homogeneous WSNs

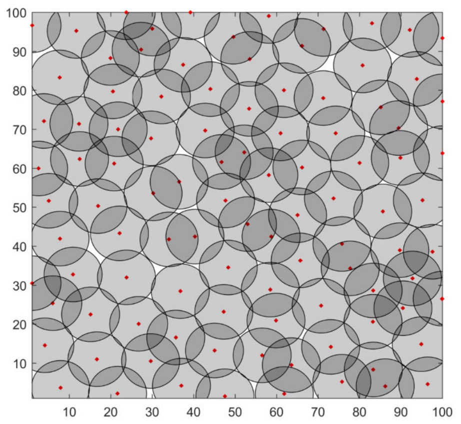
In order to analyze the performance of the IBA algorithm on the optimal deployment of homogeneous sensor nodes, a set of experimental studies has been carried out with a wireless sensor network including 100 mobile sensors. The sensing range of sensors is the same and fixed to 7m. In all the experiments, the initial positions of sensors are generated randomly inside the sensing area. The results of the IBA are compared with those of BA, qABC, and SSO algorithms. The experiments were performed 10 times for each algorithm over 1000 iterations.

IV.6.1.1 Coverage and stability comparison

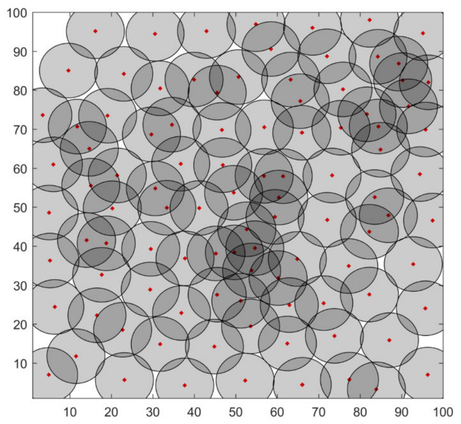
Figure IV.9 shows the best deployment distributions returned by the qABC, the SSO, the BA, and the IBA. The associated quantitative results are detailed in Table IV.3.

Table IV.3: Coverage results comparison in homogeneous case.

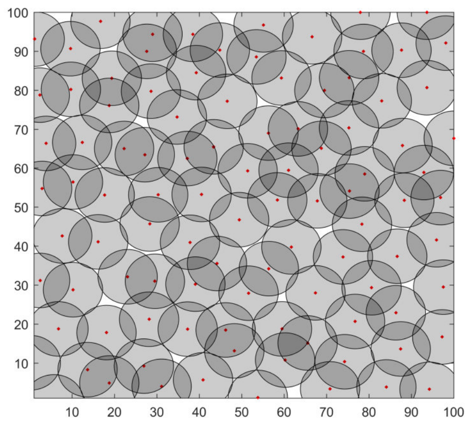
Algorithms	Best value (%)	Mean value (%)	STD
IBA	99.47	99.27	0.1332
BA	98.31	98.01	0.1740
qABC	98.93	98.67	0.2182
SSO	96.58	96.58	2.2781e-16



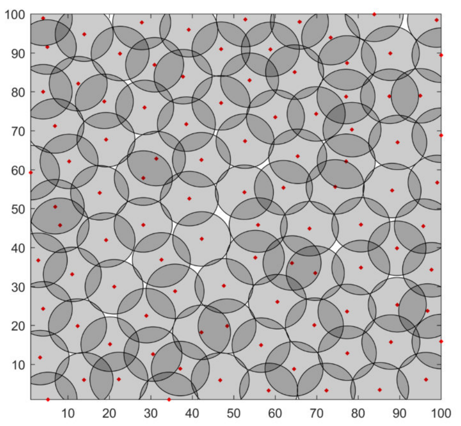
(a)



(b)



(c)



(d)

Figure IV.9: (a) final deployment of qABC, (b) final deployment of SSO, (c) final deployment of BA, and (d) final deployment of IBA.

As can be seen from Figure IV.9 and the results of Table IV.3, the SSO algorithm achieves the worst coverage value among the four competitors where it covers only 96.58% of the area of interest resulting in multiple coverage holes. The BA is ranked in third place outperforming the SSO by 1.73% where it provided an acceptable coverage value of 98.31%. On the other hand, the qABC shows good results compared to SSO and BA by achieving 98.93% of coverage and a good distribution of sensors as illustrated in Figure IV.9(a).
Figure IV.9(d) shows that the IBA can optimize the coverage effectively outperforming SSO, BA, and qABC algorithms. The superiority of the IBA has been confirmed by the best distribution of sensor nodes in the mission area with a coverage of 99.47%.
In direct comparison with BA, the coverage provided by IBA is improved up to 1.16%, which proves the superiority and the validity of the improvements introduced by the proposed technique. Besides, unlike BA, the optimal positions provided by the IBA avoid sensor clustering where the overlapping area between sensors is reduced to a minimum value. This decrease in the overlapping area eliminates both coverage redundancy and coverage holes.

In addition to the above, the IBA provides a better mean coverage value where it reaches 99.27% of coverage higher than those obtained using SSO, BA, and the qABC by 2.69%, 1.26%, and 0.60%, respectively.

When comparing the standard deviation of the algorithms, the SSO algorithm provides a much smaller value than that provided by the IBA. Still, it fails to achieve a nearest approximation of the full coverage where it covers just 96.58% of the area of interest. On the contrary, the standard deviations obtained by the BA and the qABC are considered the worst. Nevertheless, they perform better in optimizing the network coverage compared with the SSO. The IBA provides also a smaller standard deviation of 0.1332, which is considered a good value outperforming BA and qABC. Besides, it achieves the highest coverage values covering the area of interest effectively. This indicates that most of the time the IBA can optimize the coverage and obtains a good approximation of the full coverage despite the initial random positions of sensors.

IV.6.1.2 Convergence Speed Comparison

For further observing the performance of the proposed technique, the convergence of the IBA algorithm is investigated. The convergence curves of the four algorithms are illustrated in Figure IV.10. This figure shows that the IBA was successful in improving the quality of the best solution as the iteration counter increases where it reaches 99.47% of coverage. The qABC and BA algorithms provide competitive results and much faster convergence compared with SSO which has the worst convergence speed. However, we observe from the early iterations that the IBA coverage curve is always above the curves of the other algorithms, this indicates that IBA has the fastest convergence speed. Although the algorithm converges faster in the early stages, it still has the ability to develop in the later iterations. This is because the IBA employs a number of foragers to perform an extensive exploitation search in the neighborhood of the best sites. In addition, the shrinking of the neighborhood boundaries assists the algorithm in searching the areas around the optimal solution, which improves the search efficiency and allows more accurate exploitation of the best sites.

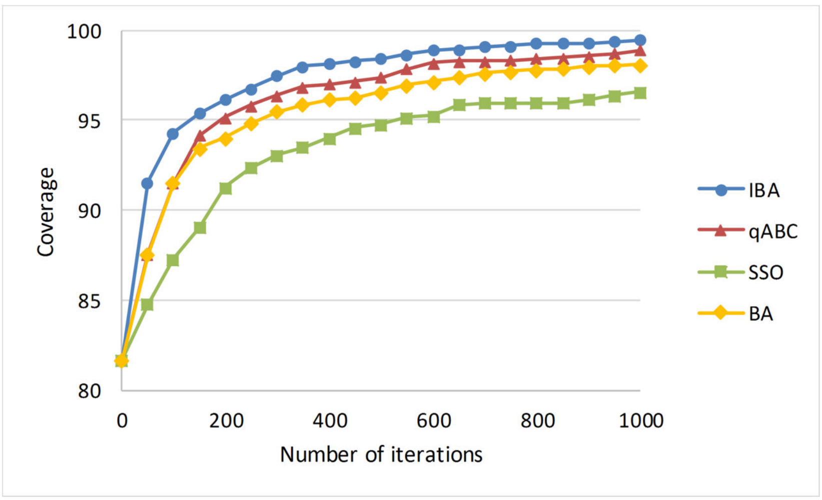


Figure IV.10: Convergence curves of qABC, SSO, BA, and IBA.

IV.6.2 Deployment of Heterogeneous WSNs

In the second part, we analyze the performance of the IBA on the deployment problem in heterogeneous wireless sensor networks. For this purpose, we used a wireless sensor network with 3 types of sensors S_1, S_2, and S_3 for which the sensing range for each sensor type is 6m, 7.5m, and 9m respectively. In our experiments, different densities for each sensor type are deployed in the area of interest, the total number of sensor nodes is varied from 20 to 100. The experiment settings are summarized in Table IV.4.

Table IV.4: Heterogeneous wireless sensor networks test cases.

Total number of sensors	Number of S_1	Number of S_2	Number of S_3
20	5	7	8
60	17	30	13
100	80	9	11

The IBA is compared to two recent related works that are also based on metaheuristics MCHSA and IFPA. Each algorithm was executed 10 times for every test case over 500 iterations. The same metrics used to evaluate the performance of the IBA for the homogeneous sensors are adopted for the case of heterogeneous sensors. The comparison results are presented in Table IV.5.

IV.6.2.1 Coverage and stability comparison

It can be seen from Table IV.5 that the IBA was able to optimize the network coverage and achieve the highest values in all the test cases. To be more specific, the IBA provides 38.91%, 90.08%, and 96.63% of coverage when deploying 20, 60, and 100 sensor nodes respectively, followed by MCHSA with a big difference from IFPA which ranked last, especially when the number of deployed sensors is large. The MCHSA provides competitive results outperforming the IFPA, this is probably due to the improvement in the local search introduced in MCHSA performed by the Hooke–Jeeves pattern search method.

Table IV.5: Coverage results comparison in heterogeneous case.

N_s	IBA		MCHSA		IFPA	
	Best (%)	Mean (%)	Best (%)	Mean (%)	Best (%)	Mean (%)
20	38.91	38.87	38.75	38.68	38.61	38.51
60	90.08	89.45	88.06	86.70	83.96	83.26
100	96.63	96.29	94.02	92.52	91.18	90.68

In comparison with the coverage produced by IFPA, the coverage provided by IBA is improved up to 0.3%, 6.12%, and 5.45% when deploying 20, 60, and 100 sensors respectively. Moreover, the coverage of the IFPA when deploying 100 sensors reaches 91.18% with a difference of 1.10% from IBA that achieved 90.08% of coverage when deploying only 60 sensors.

Therefore, under the same coverage requirements, the IBA minimize the network cost by deploying a smaller number of sensor nodes.

In further analysis, the standard deviation values for the three algorithms illustrated in Figure IV.11 shows that the IBA provides much less standard deviations than that of MCHSA and IFPA in all the test cases. This indicates that the IBA is more stable than the two other algorithms regardless of the number of deployed sensors, where it achieves coverage values close to each other in the majority of the experiments.

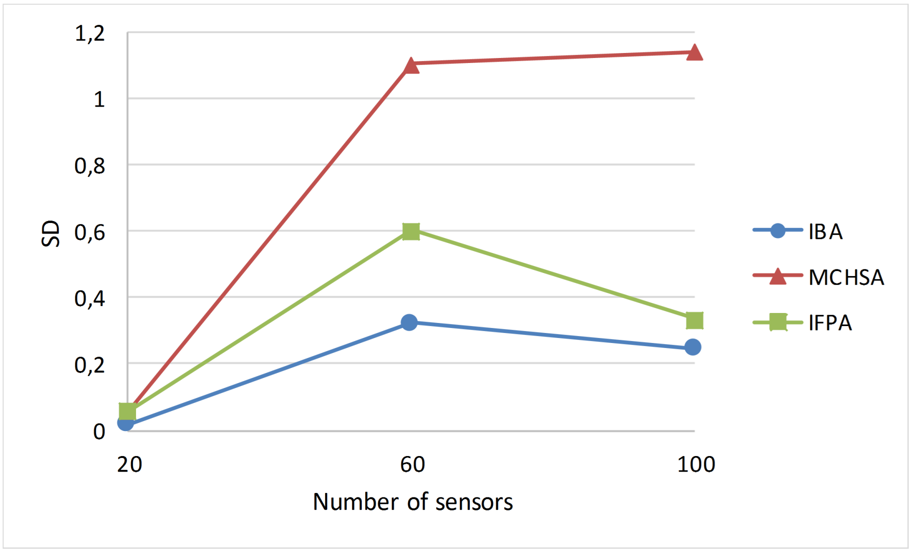


Figure IV.11: Standard deviation values for MCHSA, IFPA, and IBA.

In terms of the overall solution quality, Figure IV.12 shows that the IBA offers excellent solutions and a good distribution of sensors in the area of interest throughout the experiments outperforming the other two algorithms. The superiority of the IBA in optimizing the network coverage is confirmed by the best mean coverage values achieved during the experiments with a different number of used sensors. Moreover, it is clearly shown from the results that the IBA provides relatively good solutions in optimizing the coverage of heterogeneous WSNs.

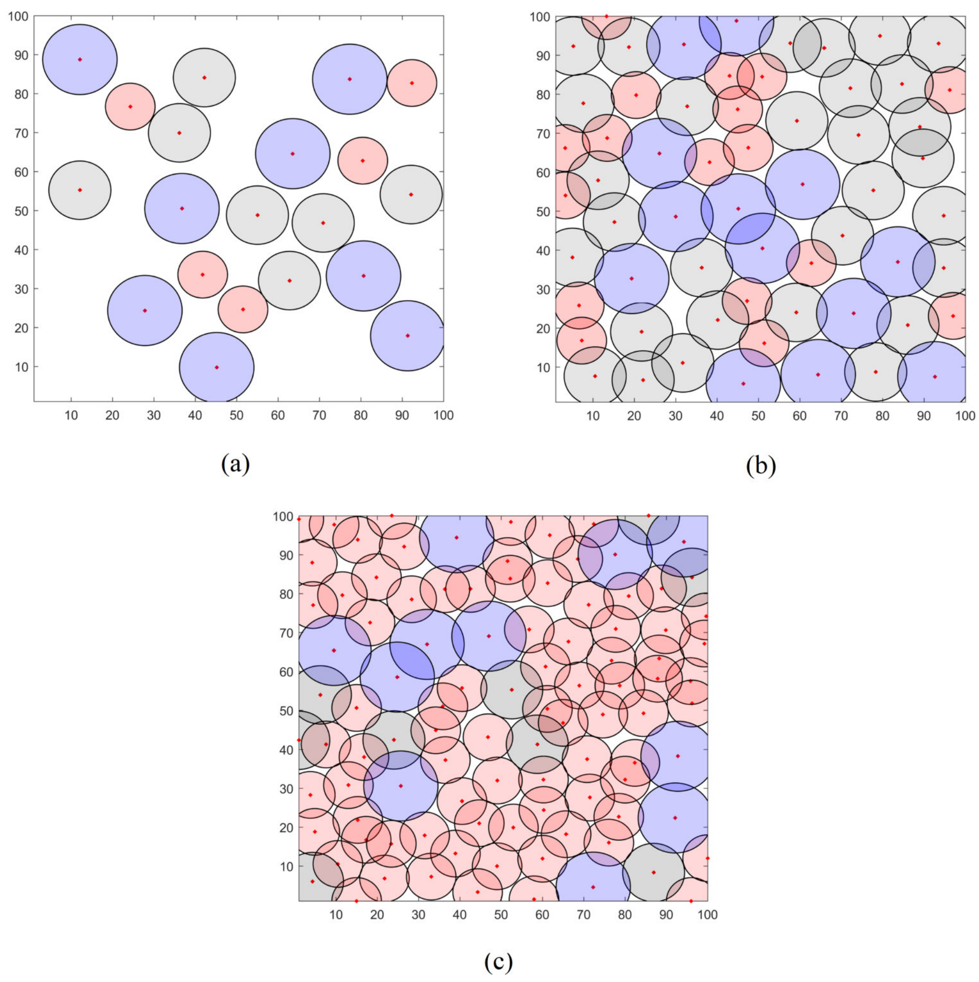


Figure IV.12: Solutions derived by IBA: (a) for 20 sensors, (b) for 60 sensors, and (c) for 100 sensors.

IV.7 Conclusion

In this chapter, we have focused on solving the deployment problem in homogeneous and heterogeneous WSNs. We have focused on maximizing the area coverage after the initial random deployment of sensors. We have introduced a novel deployment technique based on a bio-inspired algorithm called the Improved Bees Algorithm (IBA). The IBA aims to maximize the network coverage by choosing optimal positions for the sensor nodes regardless of the sensing range values.

The IBA includes two improvements namely: neighborhood shrinking and site abandonment. The two improvements introduced to the basic BA strengthens the algorithm by improving the search efficiency and allows more accurate exploitation through narrowing the local search space. The goal of the shrinking procedure is to keep the search concentrated around the best solutions in the hope of finding optimal positions of sensors that improve the coverage of the WSN. Furthermore, the site abandonment procedure strengthens the ability of the algorithm to jump out of the local optimum when there is no improvement in a particular site.

The performance of the IBA was evaluated in terms of coverage, convergence, stability and compared with other well-known bio-inspired algorithms. Comparison results have verified that the IBA can maximize the network coverage and outperform the other algorithms in terms of solution quality and stability. Besides, the IBA contributes to minimizing both coverage redundancy and coverage holes where it is capable of providing optimal deployment patterns in a very short time.

In the next chapter, we will focus on maximizing the coverage and reducing the energy consumption during the displacement of sensors toward their final locations. To achieve those objectives, we will propose a new technique for enhancing the local search of the bees algorithm.

CHAPTER V

NOVEL HYBRID ALGORITHM FOR OPTIMAL DEPLOYMENT OF WIRELESS SENSOR NETWORKS

Contents

V.1 Introduction

In this chapter, we are interested in solving the deployment problem of both mixed and mobile wireless sensor networks. For this purpose, we propose a deployment technique by hybridizing two bio-inspired algorithms, namely the Bees Algorithm (BA) and the Grasshopper Optimization Algorithm (GOA). Our proposed technique named BAGOA is designed to solve the deployment problem by maximizing the area coverage and minimizing the energy consumption during the displacement of sensors. In this chapter, we will show how these two objectives are achieved by using our novel hybrid algorithm.

As we presented in the previous chapter, the BA is an optimization algorithm that demonstrated promising results in solving many engineering problems. However, the local search process of BA lacks efficient exploitation due to the random assignment of search agents inside the neighborhoods, which weakens the algorithm's accuracy and results in slow convergence, especially when solving higher dimension problems. To alleviate this shortcoming, we propose a hybrid algorithm that utilizes the strength of the GOA to enhance the exploitation phase of the BA. We apply the BAGOA for deployment optimization with various deployment settings. BAGOA is then compared with recently proposed deployment techniques through several experiences in terms of coverage, overlapping area, average moving distance, and energy consumption.

V.2 Problem description

The deployment problem has several forms. The most popular one is constructing the WSN using a set of sensors that have the mobility feature. The deployment of mobile sensors is preferable because the designer have a complete control of all the sensors, which makes the planning of the deployment patterns much easier.

However, deploying a WSN that is composed of only mobile sensors is not always preferable due to several reasons that are directly related to the applications, such as the limited budget, because mobile sensors are expensive compared with fixed sensors. In such applications, the WSN is designed using both mobile and stationary sensors. This type of network is called a mixed WSN. In a mixed network, a number of stationary sensors are firstly deployed at random, then some mobile sensors are added to improve the network coverage. Essentially, the mobile sensors are deployed to expand the covered area and ensure full data collection.

In addition to coverage, controlling energy consumption during deployment has a significant influence on the life span of the WSN. After optimizing the deployment, the mobile sensors are required to move from their initial random positions to their final positions optimized by the algorithms. Because sensors are constrained in terms of energy, the movements of sensors should be reduced as much as possible in order to conserve energy.

In this study, we address the deployment problem of both mixed and mobile wireless sensor networks. We try to solve the deployment problem by maximizing the area coverage. In the case of mobile WSNs, the initial locations are chosen randomly in the area of interest, then our developed deployment technique attempts to determine the optimal locations of the mobile sensors. In the case of mixed WSNs, both types of sensors are deployed at random, and then our technique attempts to maximize the network coverage by changing the positions of the mobile sensors only.

We study also another major challenge in WSNs, which is minimizing energy consumption. We attempt to alleviate this challenge by minimizing the moving distance of sensors during displacement because mobility contributes a lot in depleting the energy of sensors. Therefore, how to reduce energy consumption during deployment is a major concern in our study.

In our contribution, we solve the problem of deployment optimization in WSNs by hybridizing two bio-inspired algorithms. In this hybridization, one algorithm is used to enhance the search capability of the other in order to deliver high-quality solutions when solving the deployment problem.

V.3 Proposed hybrid algorithm

In this section, our technique for solving the problem of deployment in WSNs is discussed. First, we present the idea behind our deployment technique. Then, the steps of the proposed deployment scheme are described in detail.

V.3.1 The standard Bees Algorithm shortcomings

The BA as an optimization algorithm has proven its efficiency in solving several challenging problems. It has been successfully applied to project scheduling, multiple disc clutch problem, printed circuit board assembly minimization, etc. However, the standard BA faces difficulties during optimizing a number of problems. For instance, when the problem dimension is high (as in the case de WSN deployment), it fails in producing high-quality solutions, which prevents it from achieving approximate solutions of the global optimum. The disadvantages of the BA are mainly in the exploitation phase where the repetitive unguided random search performed by the forager bees results in slow convergence and low precision.

Therefore, aiming at improving the exploitation of the BA, we propose a hybrid algorithm based on the Bees Algorithm (BA) and the Grasshopper Optimization Algorithm (GOA) named BAGOA. The GOA is a modern optimization algorithm inspired by the swarming behavior of grasshoppers. The strength of GOA lies in its high level of exploitation guided by the social interactions between all the agents in the swarm, which makes it ideal for hybridization with the BA.

V.3.2 Overview of Grasshopper Optimization Algorithm

Grasshoppers are voracious insects that eat almost all types of plants that come in their path. These small insects are considered a nightmare for farmers due to the huge damage that they inflict on agricultural crops. Usually, the grasshoppers live and eat individually in nature but most of the time millions of these insects meet and form one of the largest swarm seen in nature. Similar to other insects, the grasshopper life cycle is divided into two stages: larval and adulthood. The swarming of grasshoppers is seen in both stages but with different behaviors.

In the larval stage, the grasshoppers move slowly in the ground with very small steps because they have no wings. In contrast, adult grasshoppers form a swarm in the air where they move abruptly with larger steps.

The Grasshopper Optimization Algorithm (GOA) is a modern bio-inspired algorithm presented by Saremi et al., 2017 [185] for solving optimization problems. The characteristics of the two swarming behaviors are the main motivation of the GOA. The movement of grasshoppers in a swarm is mathematically formulated as follows:

$$X_i = S_i + G_i + A_i \tag{V.1}$$

Where X_i denotes the position of i-th grasshopper, S_i is the social interaction, G_i is the gravity force, and A_i represents the wind advection.
Among the three components in equation (V.1), the social interaction is the most important factor in the movement of grasshoppers, which can be expressed as follows:

$$S_i = \sum_{\substack{j=1 \\ j\neq i}}^{N} s(d_{ij})\widehat{d_{ij}} \tag{V.2}$$

$$d_{ij} = |x_j - x_i| \tag{V.3}$$

$$\widehat{d_{ij}} = (x_j - x_i)/d_{ij} \tag{V.4}$$

Where d_{ij} is the distance between the i-th and the j-th grasshopper, $\widehat{d_{ij}}$ is a unit vector from the i-th to the j-th grasshopper, N is the number of grasshoppers, and finally the $s()$ function represents the strength of social forces:

$$s(r) = fe^{\frac{-d}{l}} - e^{-d} \tag{V.5}$$

Where f and l represent the intensity of attraction and the attractive length scale, respectively.

Based on the distance, the space between two grasshoppers is divided into three zones: repulsion zone, attraction zone, and comfort zone (where there is neither attraction nor repulsion). Each grasshopper updates its position either by attraction or repulsion taking into consideration the positions of all grasshoppers in a swarm.
The gravity force and wind advection of the i-th grasshopper are calculated according to the following equations:

$$G_i = -g\widehat{e_g} \tag{V.6}$$

$$A_i = u\widehat{e_w} \tag{V.7}$$

Where g and u are constants, $\widehat{e_g}$ and $\widehat{e_w}$ represent the unity vector towards the center of the earth and the direction of the wind, respectively.

The mathematical model proposed so far cannot be utilized directly to perform optimization because the swarm does not converge to a particular point, a modified version of equation (V.1) is proposed in [185] to solve optimization problems as follows:

$$X_i^d = c\left(\sum_{\substack{j=1\\ j\neq i}}^{N} c\frac{ub_d - lb_d}{2} s(|x_j^d - x_i^d|)\frac{x_j - x_i}{d_{ij}}\right) + T_d \tag{V.8}$$

Where ub_d and lb_d denote the upper and the lower bounds of the d-th dimension, T_d represents the value of the d-th dimension in the best solution. The outer c is a decreasing coefficient that balance exploration and exploitation around the best solution, the inner c is utilized to shrink the comfort zone, repulsion zone, and attraction zone with respect to the number of iterations. The parameter c is computed by the following equation:

$$c = c_{max} - l\frac{c_{max} - c_{min}}{L} \tag{V.9}$$

Where l is the current iteration and L is the maximum number of iterations. Generally, the values of c_{max} and c_{min} are taken as 1 and 0.00001, respectively.

In the modified equation, the gravity force has not been considered and the wind direction has been assumed towards the target T_d. The pseudo code of the GOA algorithm is illustrated in Figure V.1.

Pseudo-code of GOA

```
Randomly initialize the position of the swarm of grasshoppers
Initialize $c_{max}$, $c_{min}$ and maximum number of iterations
Evaluate the positions of grasshoppers
T = the best grasshopper
While ( t < Maximum number of iterations ) do
    Update c using Eq.(V.9)
    For each search agent
        Normalize the distances between grasshoppers in [1, 4]
        Update the position of the current search agent by Eq.(V.8)
        Bring the current search agent back if it goes outside the
        bounderies
    End for
    Update T if there is a better solution
    increment t by 1
End while
```

Figure V.1: Pseudo code of the GOA algorithm.

V.3.3 Hybrid Bees Algorithm with Grasshopper Optimization Algorithm (BAGOA)

Randomness is an important characteristic employed by metaheuristics mainly in the exploration phase, where the search agents are subject to abrupt movements to explore different areas of the search space. In addition, it is also utilized to avoid entrapment in local solutions during optimization. However, in the exploitation phase, the search should be directed and concentrated near the best solutions to improve the solution quality and increase the chances of obtaining a good approximation of the global optimum.

The foremost matter of concern is that the BA relies on randomness in all the phases of optimization. When exploring the search space, the scouts that have found low-quality solutions are distributed uniformly at random in the search space scouting for new promising solutions. Similar to exploration, in the exploitation phase, the foragers in the BA are placed on solutions that are also generated randomly in the neighborhood of the selected solutions. Essentially, both exploration and exploitation are mainly performed using random mutations in current solutions. This random behavior is not efficient, especially during exploitation due to its influence on the convergence speed of the algorithm. The random and undirected search leads to slow convergence because the search agents were not allowed to exploit the neighborhood efficiently. Moreover, the repetitive random local search performed in the neighborhoods relatively weakens the algorithm's accuracy, stability, and success rate.

To overcome the aforementioned shortcomings, the BA is hybridized with GOA to enhance the exploitative strength of the algorithm. The purpose of proposing the hybrid algorithm is to minimize the effect of randomness in BA during exploitation to improve the algorithm's search capability and makes the search oriented toward the best solution in the neighborhood.

As aforementioned, the GOA utilizes the information of all search agents to define the next position of each one of them. Therefore, instead of randomly searching around the solution advertised by the scout, the BAGOA updates the position of each forager based on its current position, the position of the target (best solution in the neighborhood), and the position of all other foragers in the corresponding neighborhood. Initially, the foragers are distributed at random in the neighborhoods, and then the BAGOA selects the best experienced forager bee in each neighborhood as the target and adjusts the positions of the other foragers based on the social knowledge according to equation (V.8). This mechanism allows the foragers to exploit the neighborhood efficiently. Moreover, the directed search toward the best solution in each neighborhood increases the chances of achieving a good approximation of the global best, which in return leads toward superior results in terms of convergence and accuracy.

It should be noted that the parameter c in equation (V.8) reduces the movements of foragers around the best solution in the neighborhood, which is essential in exploitation. The foragers will converge towards the target as much as possible in the last steps of optimization seeking a more accurate target. The pseudo code of the proposed BAGOA algorithm is illustrated in Figure V.2.

V.4 BAGOA for deployment optimization

V.4.1 Coverage model

Similar to the work presented in the fourth chapter, we use the Boolean perception model for coverage calculation. The mathematical formulation of the Boolean perception model is detailed in chapter IV.

Pseudo-code of the BAGOA

```
Randomly initialize the position of the scouts
Initialize $c_{max}$, $c_{min}$
While t ≤ MaxIter
  1. Evaluate the fitness of the population
  2. Sort the solutions based on their fitness
  3. Update c using Eq.(V.9)
  4. Select nb solutions with the highest fitness for neighborhood search
  5. Recruit nre foragers for each of the ne elite sites selected among the nb best sites
  6. Recruit nrb foragers for each of the remaining best sites
  7. Randomly produce new solutions for the foragers in the neighborhood of the
     selected sites
  8. Calculate the distances between foragers and normalize them in [1, 4]
  9. Update the position of the foragers by Eq.(V.8)
  10. Check the feasibility of produced solutions
  11. Select the fittest solution (bee) from each patch
  12. Allocate the rest of the scouts randomly scouting for new solutions
  13. Form the new population
  14. Memorize the best solution achieved thus far
  15. increment t by 1
End
```

Figure V.2: Pseudo code of the proposed BAGOA algorithm.

V.4.2 The overlapping area

Given the fact that the sensors are homogeneous, two sensors are said to be overlapped if the distance separating them is less than twice the sensing radius. The overlapping area between the two sensors can be expressed as the sum of common grid points covered by them. Therefore, the overlapping area between the set of sensors can be computed as follows [37]:

$$Overlapp_{area} = \sum_{x=1}^{M}\sum_{y=1}^{N} overlapp(S,x,y) \tag{V.10}$$

Where

$$overlapp(S,x,y) = \begin{cases} 1, & if\ the\ grid\ point\ T(x,y)\ is\ covered\ by\ at\ least\ two\ sensors\ from\ S \\ 0, & otherwise \end{cases} \tag{V.11}$$

V.4.3 Energy consumption

In the present work, the total energy consumed by the sensing devices during movement can be computed as follows [39]:

$$E = \omega \times d_{all} \tag{V.12}$$

Where ω is the amount of energy depleted per meter of movement, d_{all} represents the average moving distance in the network, and it is expressed as follows:

$$d_{all} = \frac{\sum_{i=1}^{sn} dist(initial_i, final_i)}{sn} \tag{V.13}$$

Where sn is the number of sensors, $dist$ is a function that calculates the Euclidean distance between the initial and final positions of sensors.

V.4.4 Solution Representation

In BAGOA, the position of each food source represents a feasible solution for the problem being optimized. Assume we have $S = \{s_1, s_2, \ldots, s_i, s_n\}$ wireless sensors to be deployed in the area of interest. Each solution $X_i (i = 1, \ldots, m)$ is a vector $X_i = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$ that contains the position coordinates of all the sensors. Where x_i and y_i represent the coordinates of the i-th sensor in 2D space.

V.4.5 Initialization

The BAGOA starts by employing a number of scout bees randomly in the search space looking for food sources. As aforementioned, the position of each food source can be identified as a vector of coordinates. Therefore, given a deployment area of size $M \times N$, the initial coordinates are chosen randomly inside the bounds of the deployment area such that $0 \leq x \leq M$ and $0 \leq y \leq N$.

V.4.6 Deployment optimization

The algorithm starts by randomly deploying S sensors for each solution (food source). Then, each scout bee is assigned to one solution, which means that the number of generated solutions is equal to the number of scouts. After that, the solutions are evaluated in terms of coverage to determine the best and the elite bees. In the next step, the selected scout bees recruit a number of foragers for neighborhood search. The recruited foragers are placed on random solutions in the vicinity of the solution advertised by the scout. At this stage, the best forager in each neighborhood is selected as a target and the positions of the foragers in the corresponding neighborhood are updated according to the position update equation of GOA (equation (V.8)). If one of the foragers finds a better network coverage than the scout, that forager bee becomes the new scout and participates in the waggle dance in the next generation. The global search of the BAGOA is the same as BA where the bees that ranked last are assigned randomly in the solution space scouting for new promising solutions. The above steps are repeated until a coverage value above a predefined threshold is met or a maximum number of iterations is reached. The flow chart of the BAGOA deployment algorithm is illustrated in Figure V.3. The dotted rectangle shows the improvement introduced in this work.

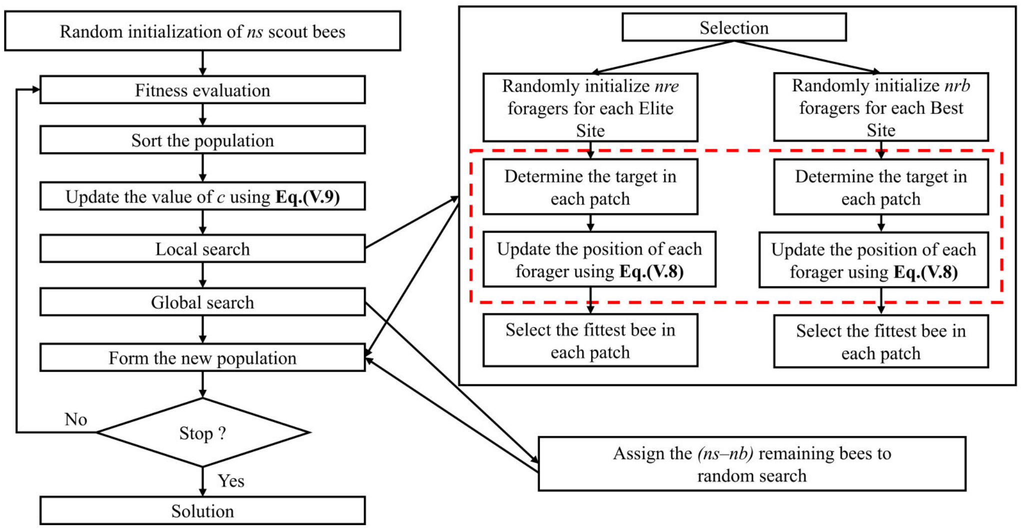

Random initialization of ns scout bees
Fitness evaluation
Sort the population
Update the value of c using Eq.(V.9)
Local search
Global search
Form the new population
No
Stop ?
Yes
Solution
Selection
Randomly initialize nre foragers for each Elite Site
Randomly initialize nrb foragers for each Best Site
Determine the target in each patch
Determine the target in each patch
Update the position of each forager using Eq.(V.8)
Update the position of each forager using Eq.(V.8)
Select the fittest bee in each patch
Select the fittest bee in each patch
Assign the (ns–nb) remaining bees to random search

Figure V.3: Flow chart of the proposed BAGOA deployment algorithm.

V.5 Experimental Results and Analysis for BAGOA

This section is designed to prove the superiority and the effectiveness of the proposed hybrid algorithm in solving the problem of deployment in WSNs. Three groups of experiments are conducted with different area sizes and sensing range values. In the first experiment, the proposed algorithm is compared with two related work algorithms namely BA and IGWO in terms of search accuracy and convergence. In the second experiment, the deployment performance of the BAGOA in a large sensing area is assessed with a different number of mobile sensors. The obtained results are compared with those of the qABC algorithm. The last experiment simulates a wireless sensor network composed of both stationary and mobile sensors. The BBO is chosen as a comparison algorithm to evaluate the results of the hybrid algorithm presented in this paper. Please note that the values written in bold represent the best results.

V.5.1 Parameter settings

Table V.1 illustrates the information about the comparison algorithms used in the set of experiments. The parameter settings of the algorithms are the same as those in the corresponding publications. The size of the population of all the comparison algorithms is set to 30. Each algorithm is executed 10 times, and the best and average results are analyzed. In all the simulations, the initial positions of mobile sensors are generated randomly inside the sensing area. The parameter values of the BAGOA used for comparison with related works are presented in Table V.2.

Table V.1: The comparison algorithms.

Algorithm	Reference	Sensor type	Coverage type
BA	[184]	Mobile	Area coverage
IGWO	[161]	Mobile	Area coverage
qABC	[152]	Mobile	Area coverage
BBO	[174]	Static and Mobile	Area coverage

Table V.2: Parameter settings of the BAGOA.

Parameter	Value
Scout bees (ns)	7
Best sites (nb)	5
Elite sites (ne)	3
Recruited bees of elite (nre)	7
Recruited bees of best (nrb)	2
Neighbourhood size (ngh)	40

V.5.2 First experiment results

In the first experiment, the BAGOA is simulated in an area of size $50m \times 50m$, the sensing radius of the sensors is the same and fixed to 5 m, and the maximum number of iterations is 200. The collected quantitative results are compared with those obtained from the simulations of the standard Bees Algorithm (BA) and the Improved Grey Wolf Optimizer (IGWO). The comparison results are illustrated in Table V.3.

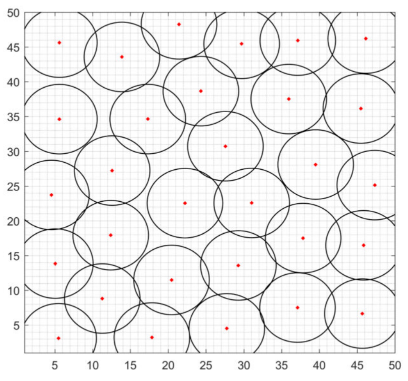
V.5.2.1 Coverage and stability comparison

Table V.3 shows that BAGOA provides superior deployment results compared with the other algorithms in all the test cases. In particular, BAGOA was the most efficient algorithm in optimizing the coverage where it achieves 87.56% and 97.52% when deploying 30 and 40 nodes respectively. Furthermore, it covers almost the entire sensing area effectively where it reaches 99.72% of coverage with 50 deployed sensor nodes higher than those obtained using BA and the IGWO by 0.72% and 0.84%, respectively. Moreover, after BAGOA optimizes the deployment, the sensors are evenly distributed as illustrated in Figure V.4 covering the sensing area effectively.

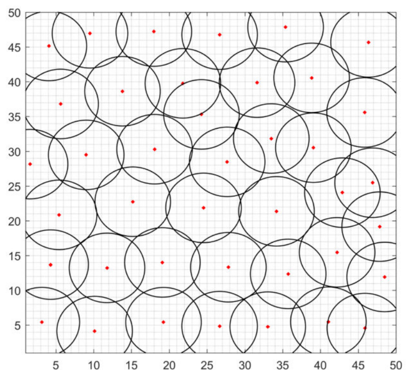
When comparing the coverage mean values of the three algorithms, Table V.3 shows that the BAGOA provides significantly better results in all the test cases. For instance, in comparison with the mean values produced by IGWO, the values provided by BAGOA are improved up to 1.54%, 2.45%, and 1.45% when deploying 30, 40, and 50 sensors, respectively. The standard deviation (Std) values in Table V.3 also prove the superiority of the proposed algorithm where it achieves the smallest standard deviations in the majority of the test cases. This indicates that most of the time the BAGOA can optimize the network coverage and obtains a good approximation of the full coverage.

Table V.3: Deployment results comparison between the algorithms in an area of size 50 m by 50 m.

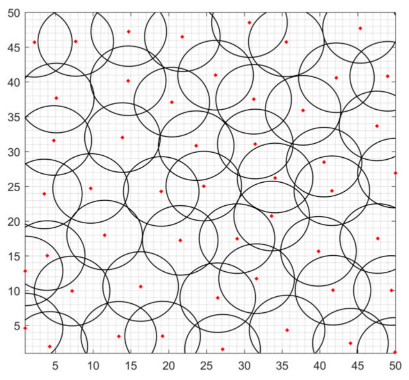
N_s	BAGOA			BA			IGWO		
	Best(%)	Mean(%)	Std	Best(%)	Mean(%)	Std	Best(%)	Mean(%)	Std
30	**87.56**	**86.45**	0.680	84.56	83.90	**0.385**	86.52	84.91	0.861
40	**97.52**	**96.74**	**0.420**	94.88	94.30	0.426	95.84	94.29	1.089
50	**99.72**	**99.39**	**0.190**	99	98.38	0.343	98.88	97.94	0.756



(a)



(b)



(c)

Figure V.4: (a) final deployment solutions derived by BAGOA: (a) for 30 sensors, (b) for 40 sensors, and (c) for 50 sensors.

V.5.2.2 Convergence Speed Comparison

In further analysis, the convergence curves of the three algorithms are shown in Figure V.5. It should be noted that the algorithms are executed with the same experimental parameters and the average of the solutions obtained in some iterations over 10 runs is compared for clarity.

As may be seen in this figure, the BAGOA significantly outperforms the BA and the IGWO in optimizing the coverage where it reaches 99.39% on average. The IGWO is ranked last, where it provides a coverage value lower than that optimized using BA by 0.44%.

In direct comparison with BA, the BAGOA tends to be accelerated faster as iteration increases. The reason why BAGOA converges fast is that the foragers search the neighborhood efficiently guided by the social interaction and oriented toward the best solution in the neighborhood instead of the repetitive blind search performed by the standard BA. In addition, we observe from the first iterations that the BAGOA coverage curve is always above the curves of the other competitors. This indicates that BAGOA has the fastest convergence speed, followed by the BA algorithm with a big difference from IGWO, which has the slowest speed. Therefore, Figure V.5 verifies the validity of the improvement proposed in this work in enhancing the convergence and the precision of the standard BA.

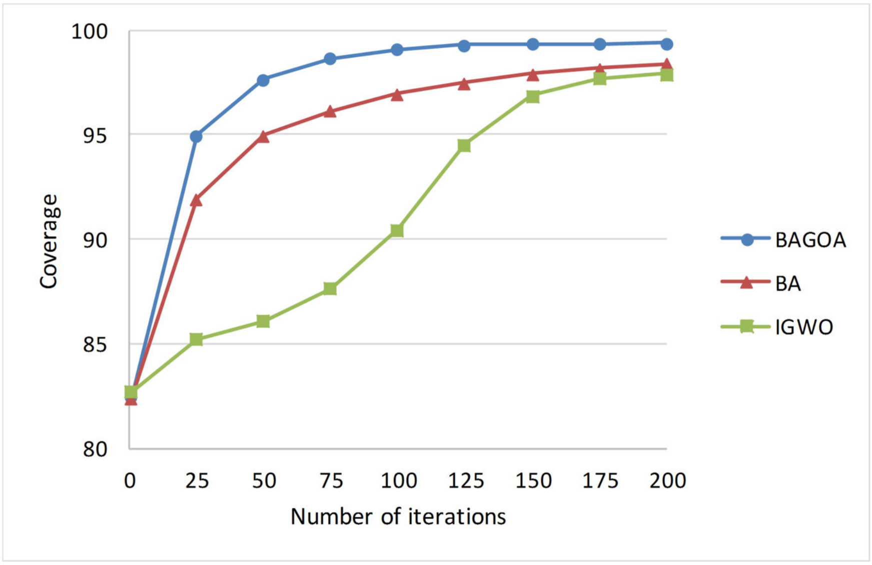


Figure V.5: Convergence curves of BA, IGWO, and BAGOA.

V.5.2.3 Energy consumption comparison

In addition to the above, the BAGOA delivers significantly better results in terms of minimizing the average moving distance and energy consumption outperforming both BA and the IGWO. As can be seen from Figures V.6 and V.7, IGWO results in fast depletion of the energy of sensors where it fails to minimize the moving distance during deployment. On the other hand, both BAGOA and BA achieved better results than IGWO with varying numbers of nodes. However, BAGOA was able to minimize the moving distance of sensors where it increases the energy saving up to 3.20%, 3.72%, and 3.65% compared with BA when deploying 30, 40, and 50 sensors, respectively. The main reason is that the movement of sensors is bounded by the neighborhood size. Unlike IGWO that move the sensors to any locations inside the entire sensing area, BAGOA avoids long distance movements by reducing the boundaries of the movement area. Therefore, each sensor will move only in the area marked by the neighborhood size. Furthermore, the shrinking of the comfort zone, repulsion zone, and attraction zone gradually reduces the movement of sensors. Consequently, this will decrease the moving distance, which in return leads to reducing the energy consumed during the displacement of sensors to their final positions. Besides, the results show that the energy consumption of BAGOA is almost stable even with the increase in the number of nodes.

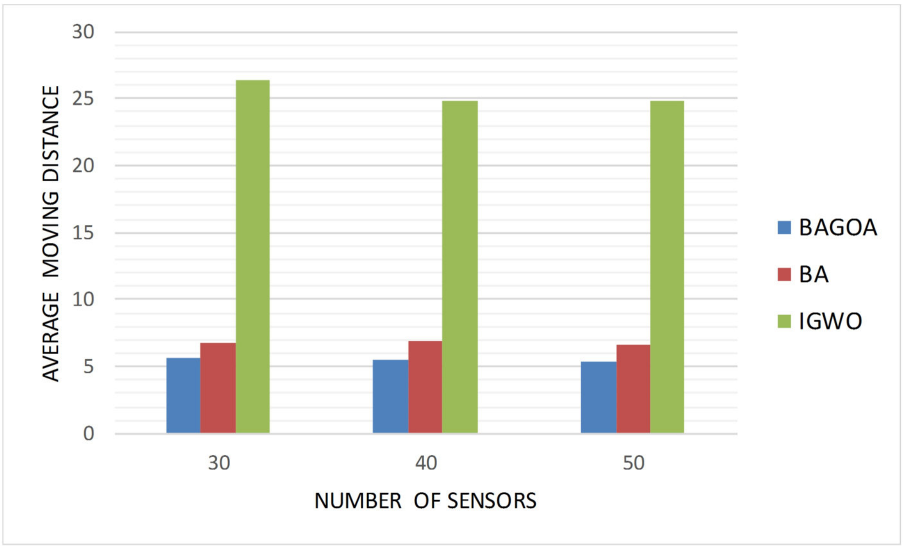


Figure V.6: Average moving distance comparison between BA, IGWO, and BAGOA.

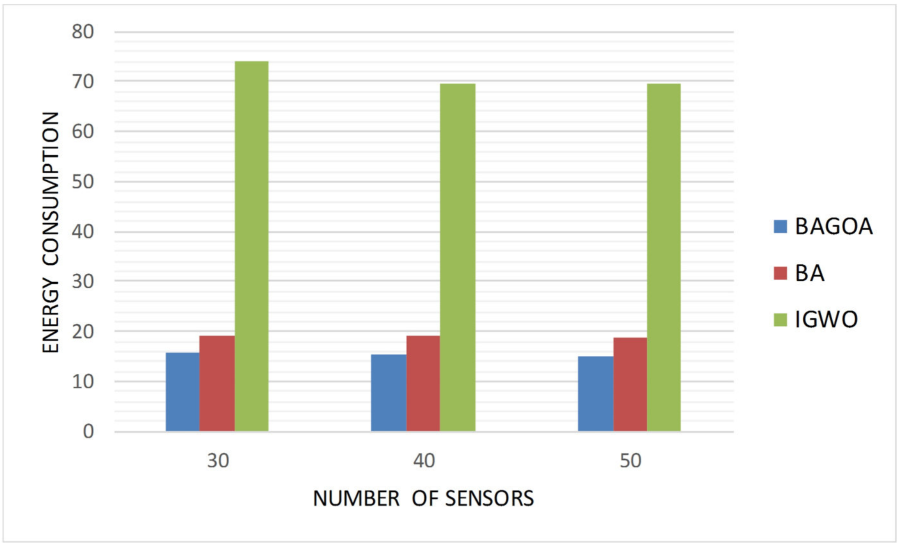


Figure V.7: Energy consumption comparison between BA, IGWO, and BAGOA.

V.5.3 Second experiment results

The second experiment was designed to observe the performance of the BAGOA in optimizing the WSN coverage when the deployment is scaled over a large geographical region. For this purpose, an area of size $100m \times 100m$ is used with a perceptual radius equals to 7 m, and a number of sensors varying between 20 and 100. In this experiment, the qABC is utilized as a comparison algorithm with the BAGOA. Each algorithm was executed 10 times with a maximum number of iterations equals to 200. The best and average results are reported in Table V.4.

V.5.3.1 Coverage and stability comparison

The results of Table V.4 show that BAGOA again performs better in optimizing the coverage, mean (average coverage), and standard deviation values, also it minimizes the average overlapping area of the network. To be more specific, the BAGOA provides 31.27%, 62%, 84.19%, and 95.47% of coverage when deploying 20, 40, 60, and 80 sensors respectively. In addition, it achieves the best approximation of the full coverage with 100 sensors, where it reaches 99.08% with a difference of 1.31% from qABC. The results indicate that the proposed algorithm maintains the best deployment performance for a large area of size $100m \times 100m$. Besides, by observing Table V.4, the BAGOA has a better mean and standard deviation values compared with the qABC in all the test cases, which confirms its stability and reliability. Moreover, it contributes to minimizing the overlapping area in the majority of the test cases except when the number of sensors is 100. It is well known that maximizing the coverage results in a decrease in the overlapping area. However, when the network is crowded and there are too many nodes deployed close to each other, the possibility for sensors to be overlapped will increase. For instance, when a coverage hole is between a set of sensors, the BAGOA requires them to overlap in order to eliminate that coverage hole. Therefore, maximizing coverage when a large number of sensors are deployed (100 sensors in our case) triggers a supplementary cost by increasing the overlapping area. Generally, as can be seen from Table V.4, the BAGOA performs better in maximizing the coverage and reducing the overlapping area between sensors to a minimum value.

Table V.4: Deployment results comparison between two algorithms in an area of size 100m by 100m.

N_s	BAGOA				qABC			
	Best(%)	Mean(%)	Std	Avg Overlapp	Best(%)	Mean(%)	Std	Avg Overlapp
20	**31.27**	**31.26**	**0.014**	**0**	31.20	31.17	0.016	1e-03
40	**62**	**61.74**	**0.134**	**0,272**	60.77	60.18	0.389	0.9780
60	**84.19**	**83.68**	**0.340**	**6,171**	82.26	80.48	1.072	8.2020
80	**95.47**	**94.67**	**0.449**	**21,542**	93.43	91.72	0.831	22.8320
100	**99.08**	**98.75**	**0.206**	41,407	97.77	97.12	0.461	**40.3460**

V.5.4 Third experiment results

In the last experiment, we analyze the performance of the BAGOA in enhancing the coverage of a mixed wireless sensor network. For this purpose, a set of simulations are conducted with a WSN containing both mobile and stationary sensors. In the simulations, a total of 100 sensors with 80 stationary sensors and 20 mobile sensors are deployed in an area of size $100m \times 100m$,

the perceptual radius is set to 7 m, and the number of iterations is 100. The BBO algorithm is chosen to make a comparison with BAGOA.
At first, the stationary sensors are deployed in the sensing area, and then the algorithms will attempt to optimize the network coverage by changing the positions of mobile sensors. To obtain a fair comparison, it should be noted that the algorithms start with the same initial positions of stationary sensors. The algorithms were simulated for 10 runs, and the collected results are presented in Table V.5.

V.5.4.1 Coverage and stability comparison

It can be seen from Table V.5 that BAGOA offers excellent coverage results throughout the experiment. More specifically, the BAGOA was successful in optimizing the coverage where it reaches 92.47%, with an increase of 23.46%. On the other hand, the BBO algorithm has an increase of 21.05%, which is 2.41% lower than BAGOA. Besides, the BAGOA outperforms the BBO in terms of average results with a difference of 3%. According to the collected results, the BAGOA is superior to the other competitor in optimizing the coverage in the case of static and mobile sensors. Moreover, the BAGOA provides a significant reduction in standard deviation value, which proves that it maintains a stable performance during the simulations.

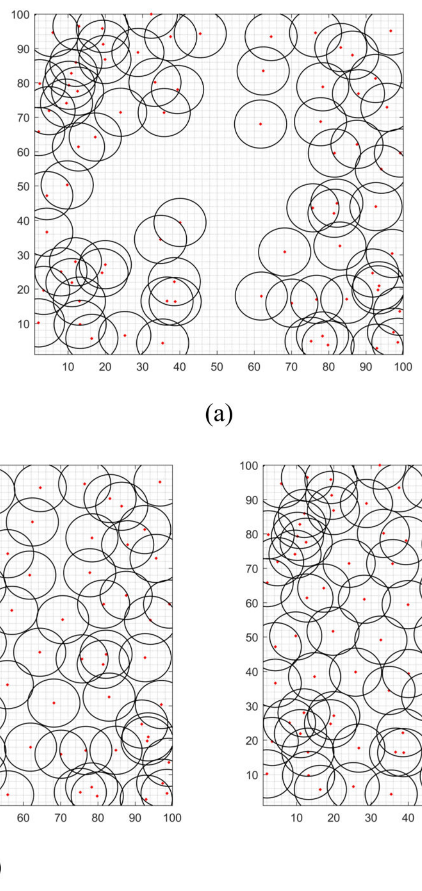
Table V.5: Deployment results comparison between two algorithms in the case of mobile and static sensors.

Algorithms	Initial coverage of stationary sensors	Best(%)	Mean(%)	Std
BAGOA	69.01	**92.47**	**91.86**	**0.321**
BBO	69.01	90.06	88.86	0.599

Figure V.8 shows the best deployment solutions of the two algorithms starting with the same initial deployment of stationary sensors. As can be seen from Figure V.8c, the BAGOA was successful in maximizing the initial coverage of the network by properly choosing a set of optimal positions for the 20 mobile sensors.

V.5.4.2 Energy consumption comparison

Regarding energy, as can be observed from Figures V.9 and V.10, BAGOA once again proves its efficiency in minimizing the amount of energy consumed during the displacement of sensors. The energy consumed by the sensors using BAGOA is less than half the energy required to move the sensors to their final positions using BBO. BAGOA offers a significant contribution to saving energy by deploying the mobile sensors in a way that optimizes the overall coverage and minimizes energy consumption, thereby, prolonging the network lifetime. The superior results are due to the shrinking of movement boundaries performed by the BAGOA that contributes to decreasing the moving distance, where it does not allow movement steps greater than the neighborhood size. Moreover, the tendency toward the best agent in the neighborhood decreases the motion rate of sensors, which leads to avoiding the fast depletion of their energy and allows the sensors to remain functional for a longer time.



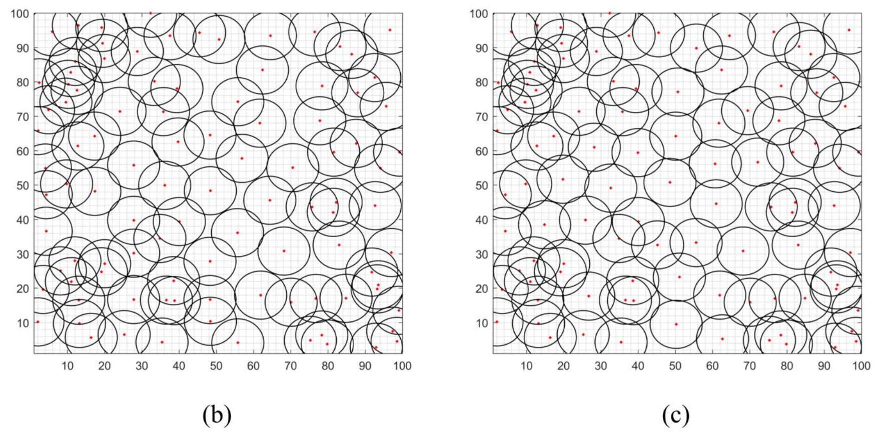


Figure V.8: (a) Initial deployment of stationary sensors, (b) Final sensor distribution of BBO, and (c) Final sensor distribution of BAGOA.

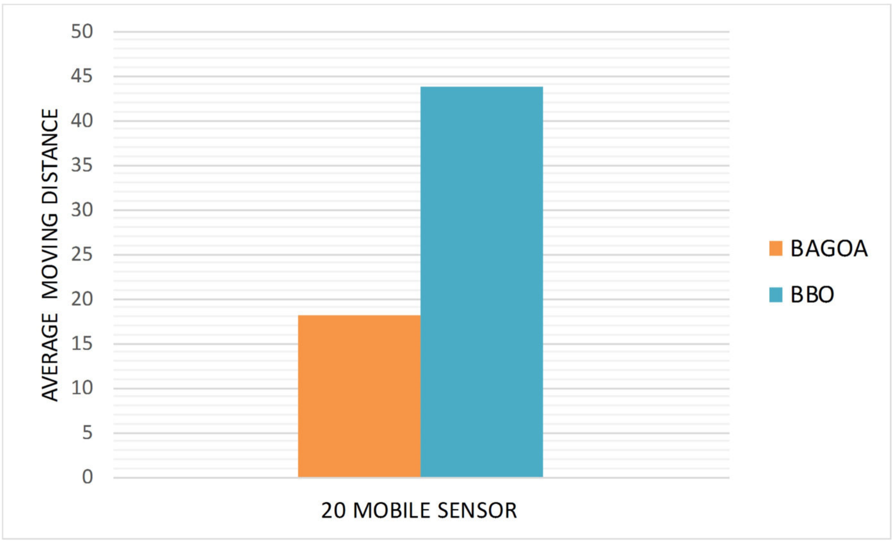


Figure V.9: Average moving distance comparison between BBO and BAGOA.

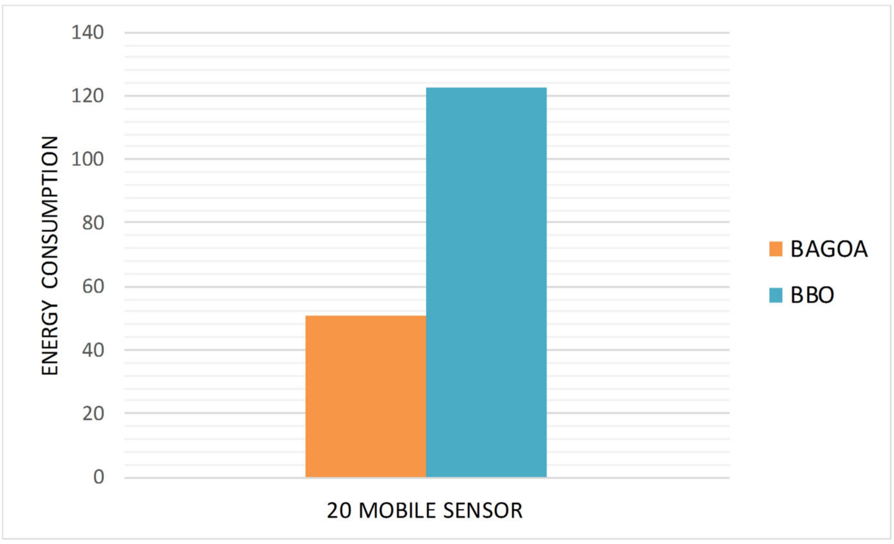


Figure V.10: Energy consumption comparison between BBO and BAGOA.

V.6 Conclusion

In this chapter, we have proposed a hybrid algorithm based on the Bees Algorithm and Grasshopper Optimization Algorithm named BAGOA to solve the problem of deployment optimization in WSNs. The proposed BAGOA algorithm utilizes the strength of the GOA to enhance the exploitative capability of the BA by searching the neighborhood more efficiently instead of the blind random search of the basic BA. By hybridizing the two algorithms, the BAGOA achieves a significant acceleration in the convergence and an increase in search accuracy. As an outcome, it led to superior results when solving the problem of deployment.

We have carried out a set of simulations and comparative studies to prove the relevance of the proposed algorithm. We compared the performance of the BAGOA with other well-known deployment algorithms in terms of coverage, convergence, and stability. The conducted comparative studies proved the efficiency of the BAGOA in optimizing the deployment of WSNs compared with other state-of-the-art algorithms where it provides excellent coverage results throughout the experiments. Furthermore, BAGOA is an energy-efficient algorithm, which contributes to maximizing the network lifetime by minimizing the amount of energy consumed by the sensors to reach their optimal positions. Besides, the superior results achieved by BAGOA in the three experiments show that it has excellent adaptability in solving different deployment problems under different experimental settings.

CONCLUSIONS AND FUTURE PROSPECTS

In this thesis, we were interested in solving one of the most important problems related to the design of WSNs. This particular problem has many names, such as sensor placement or sensor deployment, and it has a direct influence on the operations of the WSN. The importance of the deployment problem can be easily noticed by observing the growth of the volume of works presented by researchers that attempt to solve this problem. The deployment problem is popular because it affects almost all the intrinsic performance criterions of WSNs including coverage, connectivity, cost, and network lifetime.

Due to the fact that this problem was defined as an NP-hard optimization problem in most works in the literature, exact methods are not efficient to tackle this problem because they require a very long computational time to perform optimization. Bio-inspired algorithms as an alternative to exact methods have been used for obtaining the optimal solutions of various engineering design optimization problems. The use of bio-inspired algorithms becomes a major research trend in optimizing several issues related to the design of WSNs.

Our research is dedicated to addressing the problem of deployment through using bio-inspired algorithms. In the first contribution, we have introduced a deployment technique based on an Improved Bees Algorithm (IBA) for optimally deploying homogeneous and heterogeneous WSNs. The proposed technique, abbreviated IBA, adds two improvements to the original BA. These improvements aim to enhance the optimization capability of the algorithm and ensure more accurate positioning of mobile sensors. We have defined the coverage as the main objective because the essential task of any WSN is maximizing data collection. We have also given a mathematical model for the coverage, and then we applied the IBA to optimize it starting from the initial random positions of sensors. The performance of the IBA is evaluated using a varied number and type of sensors. The conducted comparative studies show that IBA delivers noticeable coverage results and good deployment patterns in the case of homogeneous WSNs. Furthermore, from the obtained results we can conclude that our strategy is suitable for deploying heterogeneous WSNs, in which multiple sensor types are used.

In the second contribution, we extended our work to consider the minimization of energy consumption when positioning the mobile sensors in optimal locations. Indeed, all applications require the guarantee of good coverage quality, however, the designers of WSNs have to give significant importance to conserving energy during deployment in order to ensure that the WSN stays operational for a longer time. To satisfy this purpose, we have proposed a novel deployment technique that takes coverage and network lifetime into consideration while planning the deployment of sensors. The proposed technique abbreviated as BAGOA is developed by hybridizing two bio-inspired algorithms namely the Bees Algorithm (BA) and the Grasshopper Optimization Algorithm (GOA). Many bio-inspired algorithms such as the BA face difficulties during optimizing a number of problems. For instance, when the problem dimension is high (as in the case de WSN deployment) they fail in producing high-quality solutions, which prevents them from achieving approximate solutions of the global optimum. To alleviate this shortcoming, we have integrated the GOA algorithm as an operator in the BA in order to improve the exploitation and the local search accuracy. The strength of GOA lies in its high level of exploitation guided by the social interactions between all the agents in the swarm, which makes it ideal for hybridization with the BA. By hybridizing the two algorithms, our strategy showed remarkable improvement in terms of coverage and energy consumption and proved that it has excellent adaptability in solving different deployment problems.

To prove the relevance of IBA and BAGOA, they have been compared to other well-known swarm intelligence algorithms, including the quick Artificial Bee Colony (qABC) algorithm, Social Spider Optimization (SSO) algorithm, Improved Grey Wolf Optimization (IGWO) algorithm, and many other recent algorithms. The results demonstrate that our techniques can deliver outstanding deployment results that surpass those delivered by these algorithms.

Although the techniques proposed in this thesis proved their efficiency in solving several versions of the deployment problem, they are far from over. They can always be improved and refined in order to support the requirements and conditions of the real world. We can think of several improvements that can be added to the present works. Some of them are stated below:

- The consideration of real-world conditions that affect the surveillance of the given area, such as sudden changes in the climate, rather than considering only ideal behavior and a set of predefined conditions.
- Extending our works to consider the uncertainty of sensor detections. In reality, the sensing zone shape of the sensor is not ideally uniform like the uniform disk-shaped model adopted in our work. Many factors affect the sensing range of sensors, such as interferences and obstacles.
- The implementation of our strategies in real-world scenarios using real sensors.
- The use of energy harvesting and renewable energy techniques to extend the lifetime of batteries, especially when deploying WSNs in inaccessible areas.
- In addition to coverage and network lifetime, several objectives must be considered by the deployment strategies, such as connectivity, cost, accuracy, and reliability.

LIST OF PUBLICATIONS

International scientific journals

Deghbouch, H., & Debbat, F. (2021). A hybrid bees algorithm with grasshopper optimization algorithm for optimal deployment of wireless sensor networks. *Inteligencia Artificial*, 24(67), 18-35.

Deghbouch, H., & Debbat, F. (2021). Improved Bees Algorithm for the Deployment of Homogeneous and Heterogeneous Wireless Sensor Networks. *International Journal of Sensor Networks*, (in press).

Deghbouch, H., & Debbat, F. (2021). An Enhanced Black Widow Optimization Algorithm for the Deployment of Wireless Sensor Networks. *International Journal of Swarm Intelligence Research*, Article Accepted in 23/11/2021.

Deghbouch, H., & Debbat, F. (2021). Hybrid Algorithm for Node Deployment with the Guarantee of Connectivity in Wireless Sensor Networks. *Informatica*, Article Accepted in 27/09/2021.

International conferences

Deghbouch, H., & Debbat, F. (2020). Dynamic Deployment of Wireless Sensor Networks using Hybrid Ant Lion Optimizer and Tabu Search. *In 2020 4th International Symposium on Informatics and its Applications (ISIA)* (pp. 1-5). IEEE.

BIBLIOGRAPHY

[1] Macker, J. P., & Corson, M. S. (1998). Mobile ad hoc networking and the IETF. *ACM SIGMOBILE Mobile Computing and Communications Review*, 2(1), 9-14.

[2] Abbas, A. M., & Kure, O. (2010). Quality of Service in mobile ad hoc networks: a survey. *International journal of ad hoc and ubiquitous computing*, 6(2), 75-98.

[3] Liu, X., Li, Z., Yang, P., & Dong, Y. (2017). Information-centric mobile ad hoc networks and content routing: a survey. *Ad Hoc Networks*, 58, 255-268.

[4] Oubbati, O. S., Atiquzzaman, M., Lorenz, P., Tareque, M. H., & Hossain, M. S. (2019). Routing in flying ad hoc networks: survey, constraints, and future challenge perspectives. *IEEE Access*, 7, 81057-81105.

[5] Padmanabhan, P., Gruenwald, L., Vallur, A., & Atiquzzaman, M. (2008). A survey of data replication techniques for mobile ad hoc network databases. *The VLDB Journal*, 17(5), 1143-1164.

[6] Sesay, S., Yang, Z., & He, J. (2004). A survey on mobile ad hoc wireless network. *Information Technology Journal*, 3(2), 168-175.

[7] Alotaibi, E., & Mukherjee, B. (2012). A survey on routing algorithms for wireless ad-hoc and mesh networks. *Computer networks*, 56(2), 940-965.

[8] Rostami, A. S., Badkoobe, M., Mohanna, F., Hosseinabadi, A. A. R., & Sangaiah, A. K. (2018). Survey on clustering in heterogeneous and homogeneous wireless sensor networks. *The Journal of Supercomputing*, 74(1), 277-323.

[9] Sangwan, A., & Singh, R. P. (2015). Survey on coverage problems in wireless sensor networks. *Wireless Personal Communications*, 80(4), 1475-1500.

[10] Rawat, P., Singh, K. D., Chaouchi, H., & Bonnin, J. M. (2014). Wireless sensor networks: a survey on recent developments and potential synergies. *The Journal of supercomputing*, 68(1), 1-48.

[11] Akyildiz, I. F., & Kasimoglu, I. H. (2004). Wireless sensor and actor networks: research challenges. *Ad hoc networks*, 2(4), 351-367.

[12] Zhu, C., Zheng, C., Shu, L., & Han, G. (2012). A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, 35(2), 619-632.

[13] Tudose, D. S., Voinescu, A., Petrareanu, M. T., Bucur, A., Loghin, D., Bostan, A., & Tapus, N. (2011, June). Home automation design using 6LoWPAN wireless sensor networks. *In 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)* (pp. 1-6). IEEE.

[14] Yu, S., Zhang, B., Li, C., & Mouftah, H. T. (2014). Routing protocols for wireless sensor networks with mobile sinks: a survey. *IEEE Communications Magazine*, 52(7), 150-157.

[15] Zhou, G. D., & Yi, T. H. (2013). The node arrangement methodology of wireless sensor networks for long-span bridge health monitoring. *International Journal of Distributed Sensor Networks*, 9(10), 865324.

[16] Kim, J. M., Park, S. H., Han, Y. J., & Chung, T. M. (2008, February). CHEF: cluster head election mechanism using fuzzy logic in wireless sensor networks. *In 2008 10th International Conference on Advanced Communication Technology* (Vol. 1, pp. 654-659). IEEE.

[17] Ari, A. A. A., Gueroui, A., Labraoui, N., & Yenke, B. O. (2015). Concepts and evolution of research in the field of wireless sensor networks. *arXiv preprint arXiv*:1502.03561.

[18] Sohraby, K., Minoli, D., & Znati, T. (2007). Wireless sensor networks: technology, protocols, and applications. *John wiley & sons.*

[19] Zeb, A., Islam, A. M., Zareei, M., Al Mamoon, I., Mansoor, N., Baharun, S., ... & Komaki, S. (2016). Clustering analysis in wireless sensor networks: The ambit of performance metrics and schemes taxonomy. *International Journal of Distributed Sensor Networks*, 12(7), 4979142.

[20] Jadidoleslamy, H. (2013). An introduction to various basic concepts of clustering techniques on wireless sensor networks. *International journal of Mobile Network Communications & Telematics (IJMNCT)* , 3(1), 1-17.

[21] Luo, H., Tao, H., Ma, H., & Das, S. K. (2010). Data fusion with desired reliability in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 22(3), 501-513.

[22] Wang, J., Gao, Y., Wang, K., Sangaiah, A. K., & Lim, S. J. (2019). An affinity propagation-based self-adaptive clustering method for wireless sensor networks. *Sensors*, 19(11), 2579.

[23] Mcgrath, M. J., & Scanaill, C. N. (2013). Sensor network topologies and design considerations. *In Sensor Technologies* (pp. 79-95). Apress, Berkeley, CA.

[24] Del-Valle-Soto, C., Mex-Perera, C., Nolazco-Flores, J. A., Velázquez, R., & Rossa-Sierra, A. (2020). Wireless sensor network energy model and its use in the optimization of routing protocols. *Energies*, 13(3), 728.

[25] Dipobagio, M. (2009). An overview on ad hoc networks. *Institute of Computer Science (ICS), Freie Universitat Berlin.*

[26] Salazar Soler, J. (2017). Wireless networks.

[27] Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer networks*, 52(12), 2292-2330.

[28] Yu, X., Wu, P., Han, W., & Zhang, Z. (2012). Overview of wireless underground sensor networks for agriculture. *African journal of biotechnology*, 11(17), 3942-3948.

[29] Aftab, M. U., Ashraf, O., Irfan, M., Majid, M., Nisar, A., & Habib, M. A. (2015). A review study of wireless sensor networks and its security. *Communications and Network*, 7(04), 172.

[30] Akyildiz, I. F., Melodia, T., & Chowdhury, K. R. (2007). A survey on wireless multimedia sensor networks. *Computer networks*, 51(4), 921-960.

[31] Di Francesco, M., Das, S. K., & Anastasi, G. (2011). Data collection in wireless sensor networks with mobile elements: A survey. *ACM Transactions on Sensor Networks (TOSN)* , 8(1), 1-31.

[32] Ramasamy, V. (2017). Mobile wireless sensor networks: An overview. *Wireless Sensor Networks—Insights and Innovations.*

[33] Fahmy, H. M. A. (2021). Protocol stack of WSNs. *In Concepts, Applications, Experimentation and Analysis of Wireless Sensor Networks* (pp. 53-66). Springer, Cham.

[34] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications magazine*, 40(8), 102-114.

[35] Almalkawi, I. T., Guerrero Zapata, M., Al-Karaki, J. N., & Morillo-Pozo, J. (2010). Wireless multimedia sensor networks: Current trends and future directions. *Sensors*, 10(7), 6662-6717.

[36] Hamid, Z., & Hussain, F. B. (2014). QoS in wireless multimedia sensor networks: a layered and cross-layered approach. *Wireless Personal Communications*, 75(1), 729-757.

[37] Messaoudi, A., Elkamel, R., Helali, A., & Bouallegue, R. (2017, June). Cross-layer based routing protocol for wireless sensor networks using a fuzzy logic module. *In 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)* (pp. 764-769). IEEE.

[38] Howitt, I., & Gutierrez, J. A. (2003, March). IEEE 802.15. 4 low rate-wireless personal area network coexistence issues. *In 2003 IEEE Wireless Communications and Networking*, 2003. WCNC 2003. (Vol. 3, pp. 1481-1486). IEEE.

[39] Safaric, S., & Malaric, K. (2006, June). ZigBee wireless standard. *In Proceedings ELMAR 2006* (pp. 259-262). IEEE.

[40] Ramya, C. M., Shanmugaraj, M., & Prabakaran, R. (2011, April). Study on ZigBee technology. *In 2011 3rd International Conference on Electronics Computer Technology* (Vol. 6, pp. 297-301). IEEE.

[41] Song, J., Han, S., Mok, A., Chen, D., Lucas, M., Nixon, M., & Pratt, W. (2008, April). WirelessHART: Applying wireless technology in real-time industrial process control. *In 2008 IEEE Real-Time and Embedded Technology and Applications Symposium* (pp. 377-386). IEEE.

[42] Kim, A. N., Hekland, F., Petersen, S., & Doyle, P. (2008, September). When HART goes wireless: Understanding and implementing the WirelessHART standard. *In 2008 IEEE International Conference on Emerging Technologies and Factory Automation* (pp. 899-907). IEEE.

[43] e Fizardo, T. P. F. (2012, January). Wibree: wireless communication technology. *In Fourth International Conference on Machine Vision (ICMV 2011): Computer Vision and Image Analysis; Pattern Recognition and Basic Technologies* (Vol. 8350, p. 83502A). International Society for Optics and Photonics.

[44] Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., ... & Culler, D. (2005). TinyOS: An operating system for sensor networks. *In Ambient intelligence* (pp. 115-148). Springer, Berlin, Heidelberg.

[45] Farooq, M. O., & Kunz, T. (2011). Operating systems for wireless sensor networks: A survey. *Sensors*, 11(6), 5900-5930.

[46] Bhatti, S., Carlson, J., Dai, H., Deng, J., Rose, J., Sheth, A., ... & Han, R. (2005). MANTIS OS: An embedded multithreaded operating system for wireless micro sensor platforms. *Mobile Networks and Applications*, 10(4), 563-579.

[47] Ahmad, I., Shah, K., & Ullah, S. (2016). Military applications using wireless sensor networks: A survey. *Int. J. Eng. Sci*, 6(6), 7039.

[48] Ali, A., Ming, Y., Chakraborty, S., & Iram, S. (2017). A comprehensive survey on real-time applications of WSN. *Future internet*, 9(4), 77.

[49] Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: an up-to-date survey. *Applied System Innovation*, 3(1), 14.

[50] Ovsthus, K., & Kristensen, L. M. (2014). An industrial perspective on wireless sensor networks—A survey of requirements, protocols, and challenges. *IEEE communications surveys & tutorials*, 16(3), 1391-1412.

[51] Xu, L., Collier, R., & O'Hare, G. M. (2017). A survey of clustering techniques in WSNs and consideration of the challenges of applying such to 5G IoT scenarios. *IEEE Internet of Things Journal*, 4(5), 1229-1249.

[52] Ramar, C., & Rubasoundar, K. (2015). A survey on data aggregation techniques in wireless sensor networks. *International Journal of Mobile Network Design and Innovation*, 6(2), 81-91.

[53] Randhawa, S., & Jain, S. (2017). Data aggregation in wireless sensor networks: Previous research, current status and future directions. *Wireless Personal Communications*, 97(3), 3355-3425.

[54] Paul, A. K., & Sato, T. (2017). Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges. *Journal of Sensor and Actuator Networks*, 6(4), 24.

[55] Sharma, A., & Sharma, S. (2016). A Comparative Review on Reliability and Fault Tolerance Enhancement Protocols in Wireless Sensor Networks. *International Research Journal of Engineering and Technology* (IRJET), 3(1), 622-626.

[56] Shyama, M., & Pillai, A. S. (2019). Fault-tolerant techniques for wireless sensor network—A comprehensive survey. *Innovations in Electronics and Communication Engineering*, 261-269.

[57] Yu, J. Y., Lee, E., Oh, S. R., Seo, Y. D., & Kim, Y. G. (2020). A Survey on Security Requirements for WSNs: Focusing on the Characteristics Related to Security. *IEEE Access*, 8, 45304-45324.

[58] Yang, Q., Zhu, X., Fu, H., & Che, X. (2015). Survey of security technologies on wireless sensor networks. *Journal of sensors*, 2015.

[59] Abdollahzadeh, S., & Navimipour, N. J. (2016). Deployment strategies in the wireless sensor network: A comprehensive review. *Computer Communications*, 91, 1-16.

[60] Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, 114, 48-70.

[61] Sioshansi, R., & Conejo, A. J. (2017). *Optimization in Engineering. Springer Optimization and Its Applications.*

[62] Binitha, S., & Sathya, S. S. (2012). A survey of bio inspired optimization algorithms. *International journal of soft computing and engineering*, 2(2), 137-151.

[63] Ibrahim, A., & Alfa, A. (2017). Optimization techniques for design problems in selected areas in WSNs: A tutorial. *Sensors*, 17(8), 1761.

[64] Rao, S. S. (2019). Engineering optimization: theory and practice. *John Wiley & Sons.*

[65] Datta, R., & Deb, K. (Eds.). (2014). *Evolutionary constrained optimization.* Springer.

[66] Karaboga, D., & Akay, B. (2011). A modified artificial bee colony (ABC) algorithm for constrained optimization problems. *Applied soft computing*, 11(3), 3021-3031.

[67] Cao, Y., & Wang, Z. (2020). Combinatorial Optimization-Based Clustering Algorithm for Wireless Sensor Networks. *Mathematical Problems in Engineering.*

[68] Mirjalili, S. (2015). The ant lion optimizer. *Advances in engineering software*, 83, 80-98.

[69] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, 46-61.

[70] Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics*, 34(5), 975-986.

[71] Droste, S., Jansen, T., & Wegener, I. (2006). Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of computing systems*, 39(4), 525-544.

[72] Mirjalili, S. (2016). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-based systems*, 96, 120-133.

[73] Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H. (2013). Metaheuristic algorithms in modeling and optimization. *Metaheuristic applications in structures and infrastructures*, 1-24.

[74] Stojanović, I., Brajević, I., Stanimirović, P. S., Kazakovtsev, L. A., & Zdravev, Z. (2017). Application of heuristic and metaheuristic algorithms in solving constrained weber problem with feasible region bounded by arcs. *Mathematical Problems in Engineering*, 2017.

[75] Dhiman, G., & Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, 165, 169-196.

[76] Mirjalili, S. Z., Mirjalili, S., Saremi, S., Faris, H., & Aljarah, I. (2018). Grasshopper optimization algorithm for multi-objective optimization problems. *Applied Intelligence*, 48(4), 805-820.

[77] Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, 114, 163-191.

[78] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249.

[79] Chen, H., Heidari, A. A., Zhao, X., Zhang, L., & Chen, H. (2020). Advanced orthogonal learning-driven multi-swarm sine cosine optimization: Framework and case studies. *Expert Systems with Applications*, 144, 113113.

[80] Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A., & Gandomi, A. H. (2021). Aquila Optimizer: A novel meta-heuristic optimization Algorithm. *Computers & Industrial Engineering*, 107250.

[81] Hussain, K., Salleh, M. N. M., Cheng, S., & Shi, Y. (2019). Metaheuristic research: a comprehensive survey. *Artificial Intelligence Review*, 52(4), 2191-2233.

[82] Mirjalili, S. (2016). Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, 27(4), 1053-1073.

[83] Kaveh, A., & Eslamlou, A. D. (2020). Water strider algorithm: A new metaheuristic and applications. *In Structures*, 25. 520-541.

[84] Shadravan, S., Naji, H. R., & Bardsiri, V. K. (2019). The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. *Engineering Applications of Artificial Intelligence*, 80, 20-34.

[85] Singh, A., Sharma, S. & Singh, J. (2021). Nature-inspired algorithms for wireless sensor networks: A comprehensive survey. *Computer Science Review*, 39, 100342.

[86] Nicklow, J., Reed, P., Savic, D., Dessalegne, T., Harrell, L., Chan-Hilton, A., ... & ASCE Task Committee on Evolutionary Computation in Environmental and Water Resources Engineering. (2010). State of the art for genetic algorithms and beyond in water resources planning and management. *Journal of Water Resources Planning and Management*, 136(4), 412-432.

[87] Jahandideh-Tehrani, M., Bozorg-Haddad, O., & Loáiciga, H. A. (2019). Application of non-animal–inspired evolutionary algorithms to reservoir operation: an overview. *Environmental monitoring and assessment*, 191(7), 1-21.

[88] Vikhar, P. A. (2016, December). Evolutionary algorithms: A critical review and its future prospects. *In 2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)* (pp. 261-265). IEEE.

[89] Verma, V. K., & Kumar, B. (2014). Genetic algorithm: an overview and its application. *International Journal of Advanced Studies in Computers, Science and Engineering*, 3(2), 21.

[90] Mirjalili, S. (2019). Genetic algorithm. *In Evolutionary algorithms and neural networks* (pp. 43-55). Springer, Cham.

[91] Das, S., & Suganthan, P. N. (2010). Differential evolution: A survey of the state-of-the-art. *IEEE transactions on evolutionary computation*, 15(1), 4-31.

[92] Singh, A., & Kumar, S. (2016). Differential evolution: an overview. *In Proceedings of Fifth International Conference on Soft Computing for Problem Solving* (pp. 209-217). Springer, Singapore.

[93] Darwish, A. (2018). Bio-inspired computing: Algorithms review, deep analysis, and the scope of applications. *Future Computing and Informatics Journal*, 3(2), 231-246.

[94] Pham, Q. V., Nguyen, D. C., Mirjalili, S., Hoang, D. T., Nguyen, D. N., Pathirana, P. N., & Hwang, W. J. (2020). Swarm intelligence for next-generation wireless networks: Recent advances and applications. *arXiv preprint arXiv*:2007.15221.

[95] Kennedy, J. (2006). Swarm intelligence. *In Handbook of nature-inspired and innovative computing* (pp. 187-219). Springer, Boston, MA.

[96] Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. *In Proceedings of the 2001 congress on evolutionary computation* (IEEE Cat. No. 01TH8546) (Vol. 1, pp. 81-86). IEEE.

[97] Primeau, N., Falcon, R., Abielmona, R., & Petriu, E. M. (2018). A review of computational intelligence techniques in wireless sensor and actuator networks. *IEEE Communications Surveys & Tutorials*, 20(4), 2822-2854.

[98] Mirjalili, S. (2019). Ant colony optimisation. *In Evolutionary Algorithms and Neural Networks*, 33-42. Springer.

[99] Sarobin, M. V. R., & Ganesan, R. (2015). Swarm Intelligence In Wireless Sensor Networks: A Survey. *International Journal of Pure and Applied Mathematics*, 101(5), 773-807.

[100] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of global optimization*, 39(3), 459-471.

[101] Sharma, A., & Chauhan, S. (2019). Target coverage computation protocols in wireless sensor networks: a comprehensive review. *International Journal of Computers and Applications*, 1-23.

[102] Priyadarshi, R., Gupta, B., & Anurag, A. (2020). Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. *The Journal of Supercomputing*, 1-41.

[103] Fellah, S., & Kaddour, M. (2017). Exact and Efficient Heuristic Deployment in WSN under Coverage, Connectivity, and Lifetime Constraints. *International Journal of Mobile Computing and Multimedia Communications*, 8(2), 27-43.

[104] Mnasri, S., Nasri, N., & Val, T. (2014). The Deployment in the Wireless Sensor Networks: Methodologies, Recent Works and Applications. *In International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks.*

[105] Chang, B. J., & Peng, J. B. (2007). On the efficient and fast response for sensor deployment in sparse wireless sensor networks. *Computer communications*, 30(18), 3892-3903.

[106] Liang, J., Liu, M., & Kui, X. (2014). A survey of coverage problems in wireless sensor networks. *Sensors & Transducers*, 163(1), 240.

[107] Mohamed, S. M., Hamza, H. S., & Saroit, I. A. (2017). Coverage in mobile wireless sensor networks (M-WSN): A survey. *Computer Communications*, 110, 133-150.

[108] Farsi, M., Elhosseini, M. A., Badawy, M., Ali, H. A., & Eldin, H. Z. (2019). Deployment techniques in wireless sensor networks, coverage and connectivity: A survey. *IEEE Access*, 7, 28940-28954.

[109] Wang, Y., Wu, S., Chen, Z., Gao, X., & Chen, G. (2017). Coverage problem with uncertain properties in wireless sensor networks: A survey. *Computer Networks*, 123, 200-232.

[110] Chen, X., Ho, Y. C., & Bai, H. (2009). Complete coverage and point coverage in randomly distributed sensor networks. *Automatica*, 45(6), 1549-1553.

[111] Khoufi, I., Minet, P., Laouiti, A., & Mahfoudh, S. (2017). Survey of deployment algorithms in wireless sensor networks: coverage and connectivity issues and challenges. *International journal of autonomous and adaptive communications systems*, 10(4), 341-390.

[112] Tao, D., & Wu, T. Y. (2014). A survey on barrier coverage problem in directional sensor networks. *IEEE sensors journal*, 15(2), 876-885.

[113] Li, L., Zhang, B., Shen, X., Zheng, J., & Yao, Z. (2011). A study on the weak barrier coverage problem in wireless sensor networks. *Computer Networks*, 55(3), 711-721.

[114] Boukerche, A., & Sun, P. (2018). Connectivity and coverage based protocols for wireless sensor networks. *Ad Hoc Networks*, 80, 54-69.

[115] Ghosh, A., & Das, S. K. (2008). Coverage and connectivity issues in wireless sensor networks: A survey. *Pervasive and Mobile Computing*, 4(3), 303-334.

[116] Zhao, J., Yağan, O., & Gligor, V. (2013). Secure k-connectivity in wireless sensor networks under an on/off channel model. *In 2013 IEEE International Symposium on Information Theory* (pp. 2790-2794). IEEE.

[117] Yetgin, H., Cheung, K. T. K., El-Hajjar, M., & Hanzo, L. H. (2017). A survey of network lifetime maximization techniques in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 19(2), 828-854.

[118] Engmann, F., Katsriku, F. A., Abdulai, J. D., Adu-Manu, K. S., & Banaseka, F. K. (2018). Prolonging the lifetime of wireless sensor networks: a review of current techniques. *Wireless Communications and Mobile Computing*, 2018.

[119] Yang, Y., Fonoage, M. I., & Cardei, M. (2010). Improving network lifetime with mobile wireless sensor networks. *Computer communications*, 33(4), 409-419.

[120] Dong, X., Cheng, L., Zheng, G., & Wang, T. (2017). Deployment cost minimization for composite event detection in large-scale heterogeneous wireless sensor networks. *International Journal of Distributed Sensor Networks*, 13(6), 1550147717714171.

[121] Priyadarshi, R., Gupta, B., & Anurag, A. (2020). Wireless sensor networks deployment: a result oriented analysis. *Wireless Personal Communications*, 113(2), 843-866.

[122] Al-Turjman, F. M., Hassanein, H. S., & Ibnkahla, M. (2013). Quantifying connectivity in wireless sensor networks with grid-based deployments. *Journal of Network and Computer Applications*, 36(1), 368-377.

[123] Park, P., Min, S. G., & Han, Y. H. (2010, December). A grid-based self-deployment schemes in mobile sensor networks. *In 2010 Proceedings of the 5th International Conference on Ubiquitous Information Technologies and Applications* (pp. 1-5). IEEE.

[124] Bartolini, N., Calamoneri, T., Fusco, E. G., Massini, A., & Silvestri, S. (2010). Push & pull: autonomous deployment of mobile sensors for a complete coverage. *Wireless Networks*, 16(3), 607-625.

[125] Tiegang, F., Guifa, T., & Limin, H. (2014). Deployment strategy of WSN based on minimizing cost per unit area. *Computer Communications*, 38, 26-35.

[126] Xiao, J., Han, S., Zhang, Y., & Xu, G. (2010, May). Hexagonal grid-based sensor deployment algorithm. *In 2010 Chinese Control and Decision Conference* (pp. 4342-4346). IEEE.

[127] Kim, Y. H., Kim, C. M., Yang, D. S., Oh, Y. J., & Han, Y. H. (2012, February). Regular sensor deployment patterns for p-coverage and q-connectivity in wireless sensor networks. *In The International Conference on Information Network 2012* (pp. 290-295). IEEE.

[128] Fortune, S. (1995). Voronoi diagrams and Delaunay triangulations. *Computing in Euclidean geometry*, 225-265.

[129] Wang, G., Cao, G., & La Porta, T. F. (2006). Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, 5(6), 640-652.

[130] Mahboubi, H., Habibi, J., Aghdam, A. G., & Sayrafian-Pour, K. (2012). Distributed deployment strategies for improved coverage in a network of mobile sensors with prioritized sensing field. *IEEE Transactions on Industrial informatics*, 9(1), 451-461.

[131] Tan, H., Wang, Y., Hao, X., Hua, Q. S., & Lau, F. C. (2010, August). Arbitrary obstacles constrained full coverage in wireless sensor networks. *In International Conference on Wireless Algorithms, Systems, and Applications* (pp. 1-10). Springer, Berlin, Heidelberg.

[132] Wu, C. H., Lee, K. C., & Chung, Y. C. (2007). A Delaunay triangulation based method for wireless sensor network deployment. *Computer Communications*, 30(14-15), 2744-2752.

[133] Zou, Y., & Chakrabarty, K. (2003, March). Sensor deployment and target localization based on virtual forces. *In IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)* (Vol. 2, pp. 1293-1303). IEEE.

[134] Li, J., Zhang, B., Cui, L., & Chai, S. (2012). An extended virtual force-based approach to distributed self-deployment in mobile sensor networks. *International Journal of Distributed Sensor Networks*, 8(3), 417307.

[135] Mougou, K., Mahfoudh, S., Minet, P., & Laouiti, A. (2012, September). Redeployment of randomly deployed wireless mobile sensor nodes. *In 2012 IEEE Vehicular Technology Conference (VTC Fall)* (pp. 1-5). IEEE.

[136] Yu, X., Liu, N., Huang, W., Qian, X., & Zhang, T. (2013). A node deployment algorithm based on van der Waals force in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 9(10), 505710.

[137] Chen, J., Li, S., & Sun, Y. (2007). Novel deployment schemes for mobile sensor networks. *Sensors*, 7(11), 2907-2919.

[138] Li, Q., Yi, Q., Tang, R., Qian, X., Yuan, K., & Liu, S. (2019). A Hybrid Optimization from Two Virtual Physical Force Algorithms for Dynamic Node Deployment in WSN Applications. *Sensors*, 19(23), 5108.

[139] Vanderbei, R. J. (2015). Linear programming (Vol. 3). *Heidelberg: Springer.*

[140] Rebai, M., Snoussi, H., Hnaien, F., & Khoukhi, L. (2015). Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks. *Computers & Operations Research*, 59, 11-21.

[141] Zhang, Y., Sun, X., & Wang, B. (2016). Efficient algorithm for k-barrier coverage based on integer linear programming. *China Communications*, 13(7), 16-23.

[142] Kukunuru, N., Thella, B. R., & Davuluri, R. L. (2010). Sensor deployment using particle swarm optimization. *International Journal of Engineering Science and Technology*, 2(10), 5395-5401.

[143] Li, Z., & Lei, L. (2009). Sensor node deployment in wireless sensor networks based on improved particle swarm optimization. *In 2009 International Conference on Applied Superconductivity and Electromagnetic Devices* (pp. 215-217). IEEE.

[144] Senouci, M. R., Bouguettouche, D., Souilah, F., & Mellouk, A. (2016). Static wireless sensor networks deployment using an improved binary PSO. *International Journal of Communication Systems*, 29(5), 1026-1041.

[145] Majid, A. S., & Joelianto, E. (2012). Optimal sensor deployment in non-convex region using discrete particle swarm optimization algorithm. *In 2012 IEEE Conference on Control, Systems & Industrial Informatics* (pp. 109-113). IEEE.

[146] Yarinezhad, R., & Hashemi, S. N. (2020). A sensor deployment approach for target coverage problem in wireless sensor networks. *Journal of Ambient Intelligence and Humanized Computing*, 1-16.

[147] Ozturk, C., Karaboga, D., & Gorkemli, B. (2011). Probabilistic dynamic deployment of wireless sensor networks by artificial bee colony algorithm. *Sensors*, 11(6), 6056-6065.

[148] Öztürk, C., Karaboğa, D., & GÖRKEMLİ, B. (2012). Artificial bee colony algorithm for dynamic deployment of wireless sensor networks. *Turkish Journal of Electrical Engineering & Computer Sciences*, 20(2), 255-262.

[149] Yadav, R. K., Gupdaa, D., & Lobiyal, D. K. (2017). Dynamic positioning of mobile sensors using modified artificial bee colony algorithm in a wireless sensor networks. *International Journal of Control Theory and Applications*, 10(18), 167-176.

[150] Aslan, S. (2019). Deployment in wireless sensor networks by parallel and cooperative parallel artificial bee colony algorithms. *An International Journal of Optimization and Control: Theories & Applications*, 9(1), 1-10.

[151] Yue, Y., Cao, L., & Luo, Z. (2019). Hybrid artificial bee colony algorithm for improving the coverage and connectivity of wireless sensor networks. *Wireless Personal Communications*, 108(3), 1719-1732.

[152] Gorkemli, B., & Al-Dulaimi, Z. (2019). On the performance of quick artificial bee colony algorithm for dynamic deployment of wireless sensor networks. *Turkish Journal of Electrical Engineering & Computer Sciences*, 27(6), 4038-4054.

[153] Li, D., Liu, W., & Cui, L. (2010, December). EasiDesign: an improved ant colony algorithm for sensor deployment in real sensor network system. *In 2010 IEEE Global Telecommunications Conference GLOBECOM 2010* (pp. 1-5). IEEE.

[154] Liu, X. (2012). Sensor deployment of wireless sensor networks based on ant colony optimization with three classes of ant transitions. *IEEE Communications Letters*, 16(10), 1604-1607.

[155] Liu, X., & He, D. (2014). Ant colony optimization with greedy migration mechanism for node deployment in wireless sensor networks. *Journal of Network and Computer Applications*, 39, 310-318.

[156] Su, H., Wang, G., Sun, X., & Yu, D. (2016). Optimal node deployment strategy for wireless sensor networks based on dynamic ant colony algorithm. *International Journal of Embedded Systems*, 8(2-3), 258-265.

[157] Deif, D. S., & Gadallah, Y. (2017). An ant colony optimization approach for the deployment of reliable wireless sensor networks. *IEEE Access*, 5, 10744-10756.

[158] Ng, C. K., Wu, C. H., Ip, W. H., & Yung, K. L. (2018). A smart bat algorithm for wireless sensor network deployment in 3-D environment. *IEEE Communications Letters*, 22(10), 2120-2123.

[159] Mohar, S. S., Goyal, S., & Kaur, R. (2021). Optimized Sensor Nodes Deployment in Wireless Sensor Network Using Bat Algorithm. *Wireless Personal Communications*, 116(4), 2835-2853.

[160] Liao, W. H., Kao, Y., & Li, Y. S. (2011). A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks. *Expert Systems with Applications*, 38(10), 12180-12188.

[161] Wang, Z., Xie, H., Hu, Z., Li, D., Wang, J., & Liang, W. (2019). Node coverage optimization algorithm for wireless sensor networks based on improved grey wolf optimizer. *Journal of Algorithms & Computational Technology*, 13, 1748302619889498.

[162] Wang, Z., & Xie, H. (2020). Wireless Sensor Network Deployment of 3D Surface Based on Enhanced Grey Wolf Optimizer. *IEEE Access*, 8, 57229-57251.

[163] Zhou, Y., Zhao, R., Luo, Q., & Wen, C. (2018). Sensor deployment scheme based on social spider optimization algorithm for wireless sensor networks. *Neural Processing Letters*, 48(1), 71-94.

[164] Fausto, F., Cuevas, E., Maciel-Castillo, O., & Morales-Castañeda, B. (2019). A Real-Coded Optimal Sensor Deployment Scheme for Wireless Sensor Networks Based on the Social Spider Optimization Algorithm. *International Journal of Computational Intelligence Systems*, 12(2), 676-696.

[165] ÖZDAĞ, R., & CANAYAZ, M. (2017). A new dynamic deployment approach based on whale optimization algorithm in the optimization of coverage rates of wireless sensor networks. *European Journal of Technique*, 7(2), 119-130.

[166] Syed, M. A., & Syed, R. (2019). Weighted Salp Swarm Algorithm and its applications towards optimal sensor deployment. *Journal of King Saud University-Computer and Information Sciences*.

[167] Tuba, E., Tuba, M., & Beko, M. (2017, April). Mobile wireless sensor networks coverage maximization by firefly algorithm. *In 2017 27th International Conference Radio elektronika (RADIOELEKTRONIKA)* (pp. 1-5). IEEE.

[168] Banimelhem, O., Mowafi, M., & Aljoby, W. (2013). Genetic algorithm based node deployment in hybrid wireless sensor networks. *Communications and Network*, 2013.

[169] Benatia, M. A., Sahnoun, M. H., Baudry, D., Louis, A., El-Hami, A., & Mazari, B. (2017). Multi-objective WSN deployment using genetic algorithms under cost, coverage, and connectivity constraints. *Wireless Personal Communications*, 94(4), 2739-2768.

[170] Hanh, N. T., Binh, H. T. T., Hoai, N. X., & Palaniswami, M. S. (2019). An efficient genetic algorithm for maximizing area coverage in wireless sensor networks. *Information Sciences*, 488, 58-75.

[171] ZainEldin, H., Badawy, M., Elhosseini, M., Arafat, H., & Abraham, A. (2020). An improved dynamic deployment technique based-on genetic algorithm (IDDT-GA) for maximizing coverage in wireless sensor networks. *Journal of Ambient Intelligence and Humanized Computing*, 1-18.

[172] Gaba, G. S., Singh, K., & Dhaliwal, B. S. (2011, December). Sensor node deployment using bacterial foraging optimization. *In 2011 International Conference on Recent Trends in Information Systems* (pp. 73-76). IEEE.

[173] Nagchoudhury, P., Maheshwari, S., & Choudhary, K. (2015). Optimal sensor nodes deployment method using bacteria foraging algorithm in wireless sensor networks. *In Emerging ICT for Bridging the Future-Proceedings of the 49th Annual Convention of the Computer Society of India CSI* (pp. 221-228). Springer, Cham.

[174] Wang, G., Guo, L., Duan, H., Liu, L., & Wang, H. (2012). Dynamic deployment of wireless sensor networks by biogeography based optimization algorithm. *Journal of Sensor and Actuator Networks*, 1(2), 86-96.

[175] Gupta, G. P., & Jha, S. (2019). Biogeography-based optimization scheme for solving the coverage and connected node placement problem for wireless sensor networks. *Wireless Networks*, 25(6), 3167-3177.

[176] Abo-Zahhad, M., Ahmed, S. M., Sabor, N., & Sasaki, S. (2014, May). Coverage maximization in mobile wireless sensor networks utilizing immune node deployment algorithm. *In 2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering* (CCECE) (pp. 1-6). IEEE.

[177] Hajjej, F., Ejbali, R., & Zaied, M. (2016). An efficient deployment approach for improved coverage in wireless sensor networks based on flower pollination algorithm. *NETCOM, NCS, WiMoNe, GRAPH-HOC, SPM, CSEIT*, 117-129.

[178] Wang, Z., Xie, H., He, D., & Chan, S. (2019). Wireless sensor network deployment optimization based on two flower pollination algorithms. *IEEE Access*, 7, 180590-180608.

[179] Tian, J., Gao, M., & Ge, G. (2016). Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm. *EURASIP Journal on Wireless Communications and Networking*, 2016(1), 1-11.

[180] Mnasri, S., Nasri, N., Van Den Bossche, A., & Val, T. (2017, September). A hybrid ant-genetic algorithm to solve a real deployment problem: a case study with experimental validation. *In International Conference on Ad-Hoc Networks and Wireless* (pp. 367-381). Springer, Cham.

[181] Panag, T. S., & Dhillon, J. S. (2019). Maximal coverage hybrid search algorithm for deployment in wireless sensor networks. *Wireless Networks*, 25(2), 637-652.

[182] Binh, H. T. T., Hanh, N. T., & Dey, N. (2018). Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. *Neural computing and applications*, 30(7), 2305-2317.

[183] Holanda, T., Almeida, T., Teixeira, P. C. M., Rodrigues, A. P. D. S. P., & Lima, R. (2019). A hybrid algorithm for deployment of sensors with coverage and connectivity constraints. *Int. J. Adv. Eng. Res. Sci.* , 6(3), 13-19.

[184] Khalaf, O. I., Abdulsahib, G. M., & Sabbar, B. M. (2020). Optimization of wireless sensor network coverage using the Bee Algorithm. *J. Inf. Sci. Eng.* , 36(2), 377-386.

[185] Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: theory and application. *Advances in Engineering Software*, 105, 30-47.

Wireless Sensor Networks (WSNs) are a type of ad-hoc network technology that has been around for more than two decades. WSNs typically consist of a number of dedicated sensors, which are fundamentally low-cost, autonomous, resource-constrained devices organized into a cooperative network to perform a common monitoring task. From their first appearance until the present day, there has been a significant effort conducted by researchers to achieve a reliable WSN design that can provide better Quality of Service (QoS) for a wide range of applications. Deployment optimization is one of the crucial issues that must be taken into consideration while designing an efficient WSN. What is meant by deployment optimization is that the sensors must be placed in strategic locations that optimize one or multiple design criteria including, coverage, connectivity, cost, and network lifetime.
In this thesis, we have addressed the problem of deployment by using bio-inspired algorithms. We proposed two deployment solutions for optimally placing homogeneous and heterogeneous WSNs. The proposed bio-inspired solutions allow the relocation of network sensors to locations that optimize coverage and energy efficiency. The first solution IBA achieves the desired objectives by eliminating both coverage redundancy and coverage holes resulted after the random deployment of heterogeneous sensors. Whereas the second solution BAGOA, which is developed by hybridizing two algorithms, namely the Bees Algorithm (BA) and the Grasshopper Optimization Algorithm (GOA), achieves high coverage and ensures low mobility during deployment in several deployment situations, even when the network is composed of both mobile and stationary sensors. The effectiveness of the proposed solutions is confirmed by the high performance recorded during the comparison with recently proposed solutions in the literature. The presented results show that IBA and BAGOA provide significantly better performance in all deployment test cases in terms of solution quality, convergence, and stability.

Keywords: WSN, Sensor deployment, Optimization, Bio-inspired algorithms, BA, GOA.

Les réseaux de capteurs sans fil (RCSFs) sont un type de technologie de réseau ad hoc qui existe depuis plus de deux décennies. Les RCSFs se composent généralement d'un certain nombre de capteurs dédiés, qui sont des dispositifs fondamentalement peu coûteux, autonomes et limités en ressources, organisés en un réseau coopératif pour effectuer une tâche de surveillance commune. Depuis leur première apparition jusqu'à nos jours, des efforts importants ont été déployés par les chercheurs pour parvenir à une conception fiable qui peut fournir une meilleure qualité de service (QoS) pour un large éventail d'applications. L'optimisation du déploiement est l'un des enjeux cruciaux qui doit être pris en considération lors de la conception d'un RCSF efficace. L'optimisation du déploiement signifie que les capteurs doivent être placés à des emplacements stratégiques qui optimisent un ou plusieurs critères de conception, notamment la couverture, la connectivité, le coût et la durée de vie du réseau.
Dans cette thèse, nous avons traité le problème du déploiement en utilisant des algorithms bio-inspirés. Nous avons proposé deux solutions de déploiement pour placer de manière optimale des RCSFs homogènes et hétérogènes. Les solutions bio-inspirées proposées permettent de déplacer les capteurs du réseau vers des emplacements qui optimisent la couverture et efficacité énergétique. La première solution IBA atteint les objectifs souhaités en éliminant à la fois la redondance de couverture et les trous de couverture résultant du déploiement aléatoire de capteurs hétérogènes. Alors que la seconde solution BAGOA, qui est développée en hybridant deux algorithmes, à savoir Bees Algorithm (BA) et Grasshopper Optimization Algorithm (GOA), atteint une couverture élevée et assure une faible mobilité pendant le déploiement dans plusieurs situations, même lorsque le réseau est composé à la fois de capteurs mobiles et fixes. L'efficacité des solutions proposées est confirmée par la performance élevée enregistrée lors de la comparaison avec les solutions récemment proposées dans la littérature. Les résultats présentés montrent qu'IBA et BAGOA offrent des performances nettement meilleures dans tous les cas de test de déploiement en termes de qualité de solution, de convergence et de stabilité.

Mots clés: RCSF, Déploiement des capteurs, Optimisation, Algorithmes Bio-inspirés, BA, GOA.

شبكات الاستشعار اللاسلكية هي نوع من تقنيات الشبكات Ad hoc التي تتواجد منذ أكثر من عقدين. تتكون شبكات الاستشعار اللاسلكية عادةً من عدد من أجهزة الاستشعار المخصصة، وهي أجهزة منخفضة التكلفة ومستقلة ومحدودة الموارد تكون منظمة في شبكة تعاونية من أجل أداء مهمة مراقبة مشتركة. منذ ظهورها لأول مرة حتى يومنا هذا، هناك جهد كبير قام به الباحثون لتحقيق تصميم موثوق لشبكات الاستشعار اللاسلكية الذي يمكنه أن يوفر جودة خدمة أفضل لمجموعة واسعة من التطبيقات. يعد تحسين النشر أحد المشكلات الحاسمة التي يجب أخذها في الاعتبار أثناء تصميم شبكة استشعار لاسلكية فعالة. المقصود بتحسين النشر هو أنه يجب وضع المستشعرات في مواقع استراتيجية تعمل على تحسين معيار تصميم واحد أو عدة معايير مثل التغطية والاتصال والتكلفة وعمر الشبكة.

في هذه الأطروحة، عالجنا مشكلة النشر باستخدام خوارزميات مستوحاة من البيولوجيا. لقد اقترحنا حلين للنشر من أجل وضع شبكات الاستشعار اللاسلكية المتجانسة وغير المتجانسة على النحو الأمثل. تسمح الحلول المقترحة المستوحاة من الحيوية بنقل مستشعرات الشبكة إلى المواقع التي تعمل على تحسين التغطية وكفاءة الطاقة. يحقق الحل الأول IBA الأهداف المرجوة من خلال القضاء على كل من تكرار التغطية وثغرات التغطية الناتجة بعد النشر العشوائي لأجهزة الاستشعار غير المتجانسة. في حين أن الحل الثاني BAGOA، الذي تم تطويره من خلال تهجين خوارزميتين، هما خوارزمية النحل و خوارزمية الجندب، يحقق تغطية عالية ويضمن تنقلًا منخفضًا أثناء النشر في العديد من الحالات، حتى عندما تكون الشبكة متكونة من مستشعرات متنقلة وثابتة. تم تأكيد فعالية الحلول المقترحة من خلال الأداء العالي المسجل أثناء المقارنة مع الحلول المقترحة مؤخرًا في الأدبيات. تظهر النتائج المقدمة أن حلينا يوفران أداءً أفضل بشكل ملحوظ في جميع حالات اختبار النشر من حيث جودة الحل والتقارب والاستقرار.

الكلمات الرئيسية: شبكات الاستشعار اللاسلكية، نشر أجهزة الاستشعار، التحسين، المستوحاة من البيولوجيا.