Optimization of sensor deployment in smart buildings

Khaoula Zaimen

Thèse de Doctorat de l'Université de Haute-Alsace

*Opérée au sein de **IRIMAS** (UHA) et **LINEACT** (CESI Strasbourg)*

Spécialité : Informatique

Soutenue publiquement le 19/06/2024, par :

Khaoula ZAIMEN

Optimisation du Déploiement de Capteurs dans les Bâtiments Intelligents

---

Devant le jury composé de :

Pr. Edward KEEDWELL	University of Exeter	Rapporteur
Pr. Bilel DERBEL	Université de Lille	Rapporteur
Pr. Karine DESCHINKEL	Université de Belfort-Montbéliard	Examinatrice
Pr. Corinne LUCET	Université de Picardie Jules Verne	Examinatrice
Pr. Lhassane IDOUMGHAR	Université de Haute-Alsace	Directeur de thèse
Pr. Abdelhafid ABOUAISSA	Université de Haute-Alsace	Codirecteur
Dr. Mohamed-El-Amine BRAHMIA	CESI de Strasbourg	Co-encadrant
MCF. Laurent MOALIC	Université de Haute-Alsace	Co-encadrant

Abstract

Smart buildings heavily depend on the integration of wireless sensor networks (WSNs) to maintain optimal functionality and operational efficacy. Consequently, any malfunctions within the sensor network directly impede the performance of these buildings. The efficiency of the WSN relies on the network topology derived from the deployment process. This process calculates the position of each sensor node within the building. An optimal WSN topology aims to maximize coverage and minimize deployment costs under the connectivity constraint.

This thesis introduces a generative and automated approach for deploying WSNs within smart buildings. The first component of the proposed approach focuses on extracting building data from a reliable digital representation of an indoor structure called Building Information Modeling (BIM). Then, it accurately models the physical layout and structural characteristics required for the deployment process. This initial preprocessing phase enables the system to address any building without necessitating prior modeling and adaptation of the deployment optimization approach for each specific building. Moreover, the deployment of WSN is modeled as a multi-objective constrained combinatorial problem. Two hybrid approaches are proposed: one based on a single-objective metaheuristic with an aggregated objective function, and another adopting a multi-objective approach. These approaches are based on classical metaheuristics and aim to determine sensor topologies that optimize coverage and cost within the target area. They comprehensively consider all building obstacles and their impact on sensor signals to produce interconnected deployment schemes. Both approaches are formulated following extensive comparative studies of various metaheuristics across diverse indoor architectures. To further enhance solution accuracy, a sensor detection zone estimation method based on fully convolutional networks is proposed. This method evaluates network coverage while considering all building obstacles. To achieve this, we have developed a dataset comprising both simple and complex indoor architectures and measured sensor signal strength in various zones using a Ray Tracing simulator. The efficiency and scalability of our deployment approaches have been evaluated and tested across different scenarios. The obtained results confirm the superiority of our methods over existing approaches, particularly in the context of medium and large-scale buildings.

Résumé

Les bâtiments intelligents dépendent fortement de l'intégration de réseaux de capteurs sans fil (RCSF) pour maintenir une fonctionnalité optimale et une efficacité opérationnelle. Par conséquent, toute défaillance au sein du réseau de capteurs entrave directement les performances de ces bâtiments. L'efficacité du RCSF repose sur la topologie du réseau dérivée du processus de déploiement. Ce processus calcule la position de chaque nœud capteur dans le bâtiment. Une topologie optimale du RCSF vise à maximiser la couverture et à minimiser les coûts de déploiement sous contrainte de connectivité.
Cette thèse présente une approche générative automatisant le déploiement de réseaux de capteurs sans fil (RCSF) au sein de bâtiments intelligents. Le premier composant de l'approche proposée se concentre sur l'extraction des données des bâtiments à partir d'une représentation numérique fiable de la structure intérieure appelée Building Information Modeling (BIM). Ensuite, il modélise de manière précise la disposition physique et les caractéristiques structurelles nécessaires au processus de déploiement. Cette phase de prétraitement initiale permet au système de traiter n'importe quel bâtiment sans nécessiter de modélisation préalable et d'adaptation de l'approche d'optimisation du déploiement pour chaque bâtiment spécifique. De plus, le déploiement du RCSF est modélisé comme un problème combinatoire contraint multi-objectif. Deux approches hybrides sont proposées : l'une basée sur une métaheuristique mono-objectif avec une fonction objectif agrégée, et une autre adoptant une approche multi-objectif. Ces approches se basent sur des métaheuristiques classiques et visent à déterminer des topologies de capteurs qui optimisent la couverture et les coûts dans la zone cible. Elles tiennent compte de manière exhaustive de tous les obstacles du bâtiment et de leur impact sur les signaux des capteurs pour produire des schémas de déploiement interconnectés. Les deux approches sont formulées à la suite d'études comparatives approfondies de diverses métaheuristiques testées sur plusieurs architectures indoor. Afin de renforcer la précision des solutions, nous proposons une méthode d'estimation des zones de détection des capteurs basée sur des réseaux entièrement convolutionnels. Cette méthode évalue la couverture du réseau tout en tenant compte de tous les obstacles du bâtiment. Pour ce faire, nous avons développé un dataset comprenant des architectures intérieures simples et complexes, et mesuré la force du signal des capteurs dans différentes zones à l'aide d'un simulateur RayTracer. L'efficacité et la scalabilité de nos approches de déploiement ont été évaluées et testées dans différents scénarios. Les résultats obtenus confirment la supériorité de nos méthodes par rapport aux approches existantes, en particulier dans le contexte de bâtiments de taille moyenne et grande.

Résumé long

Un réseau de capteurs sans fil (RCSF) représente un réseau ad hoc de nœuds capteurs homogènes ou hétérogènes qui collectent des données sur l'environnement physique et les transmettent à une station de base pour un traitement ultérieur.

L'une des applications les plus répandues des réseaux de capteurs à l'ère moderne est celle des bâtiments intelligents. Les bâtiments intelligents sont des structures équipées de caractéristiques spécifiques visant à améliorer le confort des utilisateurs et à réduire la consommation totale d'énergie. Étant donné que les bâtiments consomment environ 36 % de l'énergie mondiale, réduire cette consommation est un objectif primordial. Cela peut être réalisé grâce à l'ajustement automatique des systèmes de chauffage, de ventilation et de contrôle de la qualité de l'air (HVAC). Pour assurer ces fonctionnalités, les bâtiments intelligents nécessitent le déploiement de RCSF pour collecter des données sur la détection de présence humaine, le comptage et le suivi des personnes, la reconnaissance des activités humaines, ainsi que des facteurs environnementaux tels que la température ambiante et la qualité de l'air intérieur. Les données recueillies seront analysées et traitées pour formuler un plan d'action complet. Ce plan inclut des actions intelligentes exécutées par des actionneurs, telles que l'ajustement des systèmes d'éclairage et de HVAC, ainsi que le contrôle des verrous électroniques des portes et fenêtres.

Résoudre le problème de déploiement de RCSF dans un bâtiment intelligent implique de trouver la position optimale pour chaque capteur afin d'optimiser la couverture du réseau et d'autres métriques. L'objectif de cette thèse est de proposer des approches basées sur des métaheuristiques pour déployer des RCSFs dans des bâtiments intelligents tout en optimisant à la fois la couverture et le coût sous la contrainte de connectivité. Pour résoudre cette problématique, nous avons développé un système de modélisation de la zone cible basé sur le Building Information Modeling (BIM), qui représente une modélisation numérique détaillée du bâtiment. Le BIM intègre toutes les informations essentielles sur les éléments du bâtiment, nous permettant ainsi de réaliser une modélisation précise du bâtiment intelligent où les capteurs seront déployés. Après, nous avons modélisé le problème comme un problème combinatoire multi-objectif qui optimise à la fois la couverture et le coût sous la contrainte de connectivité. Nous avons initialement agrégé les deux objectifs en une seule fonction de fitness et proposé une approche mono-objectif pour le résoudre. L'approche mono-objectif proposée hybride la métaheuristique Grey Wolf Optimizer avec une heuristique basée sur l'arbre de Steiner pour la réduction des coûts de déploiement. Dans notre deuxième contribution, nous avons envisagé de générer un front

de Pareto de solutions non dominées de haute qualité. Pour cela, nous avons proposé une approche multi-objectif hybride, qui représente une hybridation séquentielle entre le Grey Wolf Optimizer multi-objectif et le NSGA-II. De plus, la méthode inclut une recherche locale exécutée en deux phases pour améliorer la couverture du réseau et réduire les coûts de déploiement. Notre dernière contribution consiste à proposer un nouveau modèle de détection pour les capteurs électromagnétiques dans les bâtiments intelligents. Contrairement aux modèles de détection existants dans la littérature, notre modèle proposé prend en compte toutes les caractéristiques de la zone cible où le capteur est déployé. Cela englobe l'architecture intérieure, les dimensions et les matériaux des obstacles, ainsi que leur impact sur les signaux de détection émis par les capteurs.

Modélisation du problème de déploiement des RCSF dans les bâtiments

Modélisation de la zone de déploiement basée sur la base de données BIM

L'un des facteurs clés qui impactent la conception de la topologie des RCSFs est les caractéristiques de l'environnement de déploiement, notamment sa forme et ses obstacles. Ces caractéristiques permettent de calculer les zones de déploiement qui représentent les emplacements potentiels pour les capteurs.

Une source de données complète et fiable décrivant les environnements indoor est le BIM (Building Information Modeling), qui constitue la représentation numérique d'un bâtiment tout au long de son cycle de vie [1]. Dans la littérature, l'accent mis sur l'utilisation de la base de données BIM s'est principalement porté sur la gestion de l'énergie des bâtiments, négligeant ainsi d'explorer son potentiel pour le déploiement des RCSFs [2], [3].

Dans cette thèse de doctorat, nous utilisons le fichier IFC, qui sert de base de données BIM, pour construire un modèle de la zone cible. La procédure de gestion du fichier IFC proposée se compose de deux composants principaux : le composant d'extraction de données et le composant de modélisation de la zone de déploiement. Le composant d'extraction de données agit comme un analyseur syntaxique, lisant le fichier IFC et extrayant les informations nécessaires qui représentent la disposition du bâtiment et les obstacles. Ces informations comprennent l'organisation spatiale du bâtiment (niveaux), les attributs inhérents à ces niveaux et les descriptions géométriques des composants du bâtiment.

Pour chaque niveau (étage), le composant d'extraction récupère des informations concernant les plafonds et la présence de faux plafonds, et identifie les obstacles tels que les murs, les murs-rideaux, les fenêtres, l'isolation, les portes et les poutres. Ces données IFC extraites seront traitées pour les préparer au deuxième composant, qui est la modélisation de la zone de déploiement. La fonction principale du composant de modélisation de la zone de déploiement est de faciliter le processus d'optimisation en transformant les données brutes générées par le composant d'extraction de données en un format immédiatement utilisable. Cette approche évite une boucle infinie de calcul de coordonnées spatiales continues des points de déploiement où les capteurs pourraient être placés. Pour ce faire, nous supposons que les capteurs ne peuvent être déployés que dans les faux plafonds disponibles ou dans les plafonds. Ensuite, nous organisons les points spatiaux de cette zone de déploiement (faux plafonds ou plafonds) en zones virtuelles distinctes, permettant le placement des capteurs au centre de ces zones.

Modélisation du problème de déploiement de RCSF

Dans cette thèse, nous avons modélisé le RCSF comme un ensemble de capteurs fixes. Tous les capteurs de ce réseau partagent des caractéristiques communes telles que la portée de détection, la portée de communication et la puissance de transmission. De plus, la zone cible représente un étage du bâtiment et est modélisée par le système de modélisation basé sur le BIM. Elle contient un ensemble d'obstacles statiques et un nombre fini de zones de déploiement représentant les emplacements potentiels où les capteurs peuvent être déployés. Chaque obstacle est caractérisé par sa forme, son emplacement et son matériau de construction. La zone cible comporte également un nombre fini de points cibles à couvrir par le RCSF.

Nous avons également modélisé la topologie du RCSF en utilisant un vecteur binaire de longueur égale au nombre de zones de déploiement calculées par le système de modélisation basé sur le BIM. Chaque variable de décision dans le vecteur indique la présence ou l'absence d'un capteur dans la zone de déploiement correspondante.

Réparation de la connectivité des RCSFs non connectés

Pendant la création des topologies de RCSF, la présence d'obstacles dans les bâtiments peut diviser le réseau en segments isolés de capteurs. Cette fragmentation perturbe la communication entre ces segments et entraîne une perte de données collectées par les partitions de capteurs isolées. Pour maintenir la contrainte de connectivité lors de la généra-

tion des topologies de réseaux par les méthodes d'optimisation, trois méthodes sont envisagées : l'imposition de pénalités sur les topologies non connectées, l'adaptation des opérateurs d'optimisation pour générer exclusivement des topologies interconnectées, et la réparation de la connectivité des topologies non connectées à chaque itération. Dans cette thèse, nous avons choisi la troisième approche de gestion de la connectivité, car elle surmonte les limitations des méthodes précédentes. Ainsi, pour résoudre ce problème, nous avons développé deux heuristiques de réparation de la connectivité pour corriger les topologies de capteurs en cas de multiples partitions disjointes. Les deux heuristiques sont basées respectivement sur l'algorithme de Dijkstra et sur l'arbre de Steiner minimum afin de minimiser le nombre de nœuds supplémentaires tout en préservant la topologie initiale. La première heuristique est adaptée aux zones sans obstacles ou avec peu d'obstacles, connectant les segments les plus proches à l'aide de l'algorithme de Dijkstra. La deuxième heuristique convient davantage aux zones avec des obstacles opaques. Les tests de simulation ont confirmé l'efficacité de nos méthodes par rapport aux approches existantes dans la littérature.

Approches mono-objectif pour le déploiement de RCSF dans les bâtiments intelligents

La résolution du problème de déploiement de RCSF consiste à trouver la position optimale de chaque capteur dans la zone cible tout en assurant un bon compromis entre la couverture du réseau et le coût. Un réseau entièrement interconnecté est également nécessaire afin d'éviter la perte de données collectées par des nœuds isolés. Le problème de déploiement de RCSF a été démontré comme étant un problème d'optimisation combinatoire NP-difficile [4]. Sa complexité peut augmenter significativement en raison de divers facteurs, tels que le type de zone de déploiement (indoor/outdoor), sa dimension (2D/3D), l'hétérogénéité des obstacles, l'hétérogénéité des capteurs, ainsi que le nombre de fonctions objectifs à optimiser [5]. Ainsi, l'application de méthodes exactes au problème de déploiement multi-objectif de RCSF est une tâche chronophage, caractérisée par une complexité considérable, particulièrement pour les zones de déploiement de grande et moyenne taille [5]. Par conséquent, les métaheuristiques se présentent comme des alternatives aux méthodes exactes, permettant de trouver des solutions quasi-optimales dans un temps d'execution raisonnable.

L'objectif de ce chapitre est d'étudier le comportement et de comparer les performances des métaheuristiques dans le contexte de la résolution du problème de déploiement des

RCSFs dans des environnements indoor de différentes surfaces. Les résultats de cette évaluation ont été ensuite exploités pour formuler une approche hybride mono-objective visant à améliorer davantage la qualité des topologies générées.

Metaheuristiques mono-objectif pour le déploiement de RCSF

Dans cette section, nous nous sommes basés sur l'étude présentée dans [5] et [6] pour sélectionner les algorithmes les plus populaires utilisés pour résoudre le problème de déploiement de RCSF, notamment l'algorithme génétique [7] (GA), l'évolution différentielle [8] (DE), l'algorithme de Cuckoo Search [9] (CS), l'algorithme de Particle Swarm Optimization [10] (PSO), Grey Wolf Optimizer [11] (GWO), et Bat algorithm [12] (BA) et Harmony Search [13] (HS). Les tests expérimentaux sont effectués sur dix architectures indoor, chacune présentant des superficies, des nombres, et des distributions d'obstacles hétérogènes différents. Cela permet de tester leurs performances dans différents scénarios et évalue également leur capacité de mise à l'échelle.

Les résultats obtenus montrent que les algorithmes DE, GA et HS présentent des performances similaires pour les architectures indoor de petite et moyenne taille. À l'inverse, les algorithmes PSO, CS et BA se sont révélés les moins efficaces pour tous les scénarios considérés. En revanche, le GWO a obtenu les meilleurs résultats dans tous les scénarios, démontrant ainsi son efficacité et sa capacité à résoudre le problème même dans les scénarios les plus complexes.

Méthode hybride basée sur Grey Wolf Optimizer pour le déploiement de RCSF

En se basant sur les résultats obtenus dans l'etude comparative qui ont validé l'efficacité et la mise en échelle du GWO par rapport aux autres algorithmes, nous proposons une approche novatrice basée sur le GWO binaire pour résoudre le problème du déploiement de RCSF dans les environnements indoors. L'approche proposée intègre une heuristique basée sur l'arbre de Steiner afin de réduire les coûts de déploiement tout en maintenant la connectivité et le taux de couverture de la topologie en question en tant que contraintes. La tâche de déterminer quels capteurs peuvent être désactivés dans le réseau est complexe, car la désactivation de capteurs inappropriés peut entraîner un partitionnement du réseau ou une diminution de la couverture. Notre heuristique basée sur l'arbre de Steiner vise à identifier un sous-ensemble de capteurs au sein d'une topologie RCSF qui devrait

être préservé pour garantir la couverture des mêmes points cibles tout en satisfaisant la contrainte de connectivité. Les performances de la méthode développée ont été évaluées sur six scénarios dérivés de trois bases de données BIM et comparées à quatre variantes du GWO proposées dans la littérature. Les résultats des tests montrent que notre méthode développée surpasse toutes les autres variantes du GWO, et cela pour tous les scénarios.

Approches multi-objectifs pour le déploiement de RCSF dans les batiments intelligents

Dans notre deuxième contribution, nous avons traité les objectifs conflictuels de déploiement simultanément. L'avantage de ces méthodes par rapport aux approches agrégées réside dans leur capacité à fournir aux décideurs un ensemble de solutions optimales de Pareto qui couvrent plus largement l'espace de recherche. À cette fin, dans ce chapitre, nous proposons d'examiner les performances des algorithmes évolutionnaires et des algorithmes multi-objectifs basés sur les essaims pour traiter le déploiement des capteurs, en tenant compte à la fois de la couverture et du coût de déploiement simultanément. Ensuite, nous introduisons un algorithme hybride multi-objectifs qui combine MOGWO (Multi-Objective Grey Wolf Optimizer) avec NSGA-II (Non-dominated Sorting Genetic Algorithm II) pour améliorer le processus d'exploitation. De plus, une recherche locale en deux étapes est introduite avec une certaine probabilité pour améliorer davantage la qualité des topologies produites.

Metaheuristiques multi-objectif pour le déploiement de RCSF

Pour cette étude comparative, nous avons sélectionné deux algorithmes évolutionnaires multi-objectifs : Non-dominated Sorting Genetic Algorithm (NSGA-II) et Strength Pareto Evolutionary Algorithm (SPEA-II), ainsi que des métaheuristiques basées sur les essaims qui sont : Multi-objective Particle Swarm Optimization et Multi-Objective Grey Wolf Optimizer (MOGWO), pour résoudre le problème de déploiement des RCSF et évaluer leurs performances dans dix scénarios indoors. Les résultats obtenus ont démontré l'efficacité et la supériorité de NSGA-II et de MOGWO en termes d'indicateurs de l'hypervolume et de distance générationnelle inversée (IGD).

LHNSGA-II : NSGA-II hybride avec recherche locale pour le déploiement de RCSF

Sur la base des résultats de l'étude comparative précédente, nous avons introduit un nouvelle approche hybride, appelé LHNSGA-II, qui combine NSGA-II avec MOGWO et une recherche locale. LHNSGA-II utilise une approche hybride séquentielle, où MOGWO s'exécute pendant les premières itérations afin de produire une archive de solutions non dominées de haute qualité. Ensuite, cette archive est intégrée à la population initiale de NSGA-II, qui continue son exécution pour les itérations restantes. Cette intégration aide NSGA-II à améliorer sa capacité d'exploitation tout en préservant la diversité de la population. La recherche locale proposée vise à améliorer davantage la couverture et les coûts de déploiement des topologies de RCSFs résultantes. Elle comprend deux phases, la phase initiale se concentre sur l'augmentation de la couverture du réseau, tandis que la phase suivante vise à réduire les coûts de déploiement. Les résultats obtenus montrent que LHNSGA-II surpasse à la fois MOGWO et NSGA-II, atteignant des solutions de haute qualité en termes d'hypervolume et d'IGD en moins d'itérations.

Estimation de la zone de détection des capteurs électromagnétiques dans les bâtiments intelligents

Ce chapitre introduit un nouveau modèle de détection pour les capteurs électromagnétiques dans les bâtiments intelligents. Le modèle de détection proposé peut être intégré aux approches de déploiement pour améliorer davantage leur précision. L'objectif est d'évaluer la puissance des signaux de détection dans la zone de déploiement et d'identifier à la fois les zones couvertes et non couvertes par un capteur donné, tout en tenant compte de l'hétérogénéité des obstacles.

Modèle de détection proposé dans les bâtiments intelligents

Le processus de détection est une tâche de régression par pixel qui consiste à déterminer la puissance du signal de détection réfléchi de chaque emplacement dans la zone considérée. En s'appuyant sur ce principe, notre modèle de détection proposé est basé sur le modèle SDU-Net (U-Net utilisant des convolutions dilatées empilées), une variante du modèle U-net adaptée à la régression par pixel. L'objectif de notre modèle de détection est d'identifier les zones couvertes et non couvertes dans la zone de déploiement. Cela est

réalisé en générant une carte des signaux illustrant la puissance des signaux de détection réfléchis de toutes les positions dans la zone de déploiement. Ainsi, les zones où la puissance du signal de détection dépasse le seuil de sensibilité du capteur sont considérées comme couvertes, sinon le signal est interprété comme du bruit indiquant que les zones correspondantes ne sont pas couvertes.

L'entrée du modèle de détection proposé décrit la zone de propagation où le capteur est déployé. Elle comprend quatre canaux : permittivité, conductivité, distance et Line of sight (LoS). La sortie de notre modèle de détection représente une carte de puissance du signal, modélisée par une matrice de la même taille que le tenseur d'entrée. Chaque cellule de la sortie représente une position dans l'environnement indoor et contient la valeur de la puissance du signal de détection réfléchi de cette position.

Génération de dataset

Les solveurs de RayTracing 3D disponibles sur le marché ne sont pas capables de simuler correctement un capteur électromagnétique en une seule phase de simulation. Ces solveurs effectuent généralement des simulations impliquant la propagation des signaux d'une antenne émettrice à un ensemble d'antennes réceptrices. Une fois la simulation de propagation terminée, le solveur calcule les caractéristiques essentielles des signaux reçus, telles que la force du signal et le temps d'arrivée. Cependant, lorsqu'il s'agit de capteurs électromagnétiques, notre intérêt porte sur la force des signaux de détection réfléchis provenant de chaque emplacement de la zone de propagation. Ces informations sont cruciales pour identifier les emplacements couverts (où le capteur peut détecter une présence humaine) et ceux qui ne le sont pas. Pour simuler avec précision le principe de fonctionnement complet d'un capteur électromagnétique, nous avons suivi un processus en deux étapes. Tout d'abord, ous avons simulé la propagation des signaux de détection dans toutes les directions au sein de la zone de déploiement indoor. Ensuite, nous avons simulé la réflexion de ces signaux vers le capteur. Toutes les simulations ont été réalisées à l'aide du logiciel Wireless InSite.

Pour ce dataset, nous avons généré 13 environnements indoor basiques. Dans chacun d'eux, nous avons varié les matériaux de construction parmi le placoplâtre, le bois, le béton, le verre et la brique, qui sont des matériaux couramment utilisés dans la construction des bâtiments. De plus, nous avons créé 23 environnements indoor complexes avec des agencements architecturaux diversifiés. Pour chaque architecture indoor, nous avons configuré une antenne émettrice (Tx) et un ensemble d'antennes réceptrices (Rx). Le nombre d'antennes Rx variait en fonction de l'agencement architectural. Après, nous avons

effectué une simulation en deux étapes pour chaque scénario, où nous avons reconfiguré toutes les antennes afin de simuler le capteur et transmettre les signaux de détection reçus. Par la suite, nous avons généré des cartes de la puissance du signal reçu pour identifier les emplacements couverts par le capteur.

Le dataset a été augmenté en appliquant des rotations et des retournements sur les entrées existantes. En fonction de l'environnement indoor, diverses opérations peuvent être appliquées, notamment une rotation de 90 degrés, une rotation de 180 degrés, une rotation de 270 degrés, un retournement vers la gauche, un retournement vers le bas, un retournement vers la gauche suivi d'une rotation de 90 degrés, et un retournement vers le bas suivi d'une rotation de 90 degrés.

Intégration du modèle de détection proposé avec l'approche générale de déploiement du RCSF

Le modèle de détection développé peut être intégré aux approches de déploiement proposées pour évaluer la couverture du réseau. Ces approches génèrent des schémas de déploiement de RCSF, représentés par des vecteurs binaires qui assignent les capteurs à des zones de déploiement désignées. Pour calculer la couverture pour une topologie donnée de RCSF, le processus commence par examiner le vecteur binaire pour identifier les positions de tous les capteurs, telles que déterminées par la méthode d'optimisation. Ensuite, pour chaque capteur, les étapes suivantes sont effectuées : identifier la position du capteur par ses coordonnées dans le bâtiment et extraire les caractéristiques des obstacles du bâtiment qui se trouvent dans la zone de détection maximale du capteur. Cela inclut l'architecture intérieure, les formes et l'épaisseur des obstacles, ainsi que leurs matériaux de construction. Les données extraites du bâtiment, issues d'un fichier IFC, sont ensuite traitées par des modules d'interprétation et d'organisation des données géométriques pour convertir ces informations en données exploitables. À partir de là, les canaux du tenseur d'entrée correspondant (canal de permittivité, canal de conductivité, canal de distance et canal de line of sight) sont générés pour la zone donnée. Enfin, la carte de puissance du signal de détection est générée en appliquant le modèle entraîné au tenseur d'entrée. Tous les points cibles à l'intérieur de la couverture du capteur sont identifiés en comparant la puissance du signal prédite avec le seuil de sensibilité du capteur, déterminant ainsi si chaque point cible est considéré comme couvert ou non par le capteur.

Dédicace

À ceux qui me sont chers,
Khaoula

Remerciement

C'est avec un immense plaisir que j'exprime mes sincères remerciements à toutes les personnes qui ont contribué de près ou de loin à la réussite de cette thèse de doctorat.

Tout d'abord, je souhaite remercier chaleureusement mon directeur de thèse, Pr. Lhassane IDOUMGHAR, ainsi que mon co-directeur, Pr. Abdelhafid ABOUAISSA, de m'avoir offert l'opportunité de réaliser ce travail. Leurs précieux conseils et leur soutien constant tout au long de ma thèse ont été inestimables.

Je tiens également à exprimer ma reconnaissance envers mes co-encadrants, Dr. Mohamed-El-Amine BRAHMIA et Dr. Laurent MOALIC, pour leurs orientations, leurs motivations et leurs conseils constructifs qui ont enrichi mes recherches.

Un grand merci aux membres du jury, Pr. Edward KEEDWELL, Pr. Bilel DERBEL, Pr. Karine DESCHINKEL, et Pr. Corinne LUCET, pour leur intérêt pour ce travail et pour avoir accepté de faire partie du jury et d'examiner mes travaux de thèse.

Je souhaite également exprimer ma gratitude à tous mes collègues de l'équipe OMEGA de l'IRIMAS et de l'équipe LINEACT du CESI Strasbourg pour les moments de partage et les discussions fructueuses.

Enfin, je souhaite adresser un remerciement tout particulier à ma chère famille et à toutes mes amies, dont le soutien constant et l'encouragement m'ont été précieux tout au long de ce parcours.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

INTRODUCTION

Context of the thesis

Wireless sensor network (WSN) represents a promising paradigm of networking and computing. It is an ad-hoc network of homogeneous or heterogeneous sensor nodes that sense physical environments and transmit data to a base station for further processing. In big sensor networks covering vast regions, such as in the context of smart cities, the network may incorporate some sink nodes characterized by an extended communication range. These sink nodes serve as gateways to relay the sensed data to the central base station. Diverse sorts of sensor nodes are available on the market, including light sensors, temperature sensors, pollution sensors, occupancy sensors, gas sensors, etc. Therefore, they are presently employed in many fields, namely military monitoring and tracking, health care, industry, environmental monitoring, smart agriculture, and smart buildings. Each application domain has its specifications and requirements for the quality of service (QoS) metrics that should be met when deploying the sensor network.

Contrary to traditional networks that focus on Quality of Service (QoS) metrics like throughput, delay, latency, and packet loss, Wireless Sensor Networks (WSNs) prioritize alternative criteria. These include coverage, connectivity, energy consumption, network lifespan, and deployment cost [14]. Each application prioritizes certain metrics based on its requirements. For instance, applications targeting hostile environments need to optimize network lifespan and energy consumption since the recharging of depleted nodes is a very complicated task, and in several scenarios, its fulfillment appears to be impossible. Conversely, such metrics hold lower importance for applications designed for smart buildings or home automation. However, all WSN applications need to optimize network coverage

and guarantee connectivity to prevent data loss.

One of the most widespread applications of sensor networks in the modern era is smart buildings. Smart buildings are structures equipped with specific features aimed at enhancing user comfort and reducing total energy consumption. Given that buildings consume approximately 36% of global energy [15], reducing energy usage is a primary focus. This can be achieved through the automatic adjustment of heating, ventilation, and air quality monitoring (HVAC) systems. To ensure these functionalities, smart buildings necessitate the deployment of WSNs to collect data on human presence detection, people counting and tracking, recognition of human activities, as well as environmental factors like ambient temperature and indoor air quality. The aforementioned data will be subject to analysis and processing to formulate a comprehensive action plan. This plan includes intelligent actions executed by actuators such as the adjustment of lighting and HVAC systems and the control of electronic door and window locks.

Thesis challenges

Designing the deployment scheme of a WSN in a smart building requires finding the best position for each sensor so that the overall coverage of the network will be maximized. Along with optimizing network coverage, minimizing the number of sensor nodes is imperative to mitigate deployment costs, particularly when the WSN involves expensive nodes such as electromagnetic sensors [5]. Furthermore, the network connectivity should be maintained for the generated topologies in order to prevent having isolated sensors and thus avoid any potential data loss. This turns WSN deployment into a constrained multi-objective problem that simultaneously optimizes network coverage and cost under the connectivity constraint.

Conventional approaches to WSN deployment involve a manual placement of sensors, with an iterative assessment of the requisite QoS metrics until a satisfactory deployment is attained [16]. These methodologies necessitate human intervention, and their outcomes may deviate significantly from optimal solutions. Geometric-based strategies have also been applied in the literature for deterministic deployments, however, their efficacy is constrained particularly in scenarios where the geometry of the deployment area exhibits irregular characteristics. Moreover, since the WSN deployment problem is NP-hard [4], finding the optimal solution with an exact method is a very time-consuming task that requires powerful computational resources, especially in the case of large-scale networks.

Given the multi-objective nature of the problem, decision-makers often require a set of so-

lutions that represent a trade-off between conflicting objectives. As a result, current WSN deployment approaches employ metaheuristics as they can generate a set of near-optimal solutions within a reasonable amount of time. Nonetheless, the proposed metaheuristic-based deployment approaches in the literature present several drawbacks. Firstly, while metaheuristics can be applied to solve a wide range of optimization problems, some may be more suitable for specific problems than others [17]. Yet, the proposed solutions often lack adequate justification for the choice of metaheuristics, as they are typically tested in oversimplified target areas of small and medium sizes. This limitation makes it challenging to assess their real performance in complex scenarios and their scalability when dealing with larger target areas. Additionally, the modeling of WSNs is often oversimplified. According to our survey in [5], more than 68% of reviewed metaheuristic-based approaches use a simple binary sensing model, while 63% do not specify the communication range or rely on a simple binary communication model.

Typically, deployment approaches for WSN in existing literature lack specification regarding the nature of the target area whether it is indoor or outdoor. The limited number of approaches specifically designed for indoor environments is associated with a set of constraints arising from challenges that include:

- **Lack of reliable data source describing the building:** A major challenge encountered in WSN deployment approaches stems from the absence of a comprehensive and reliable description of the target area. Approaches based on simplistic 2D representations often overlook crucial building-related information, notably the characteristics of building obstacles and their influence on sensor signal propagation. Indeed, Smart buildings present a complex deployment environment characterized by diverse obstacles of varying dimensions, thickness, and construction materials. These factors significantly impact the detection and communication signals of sensors, thereby reducing their effective detection and communication ranges. Hence, it is crucial to consider all building characteristics during sensor design to accurately assess network performance and avoid disparities between simulated and real-world network efficiency.

- **Need for adaptability to new indoor environments:** The deployment approach for WSNs necessitates the modeling of complex target areas where sensors will be deployed. This entails conveying all building characteristics to the deployment optimization module for the efficient computation of sensor positions. Existing deployment approaches typically start directly with the deployment procedure. This requires frequent re-adaptations and complex remodeling each time a new target area

is considered. Consequently, the development of an automated solution capable of modeling any given target area based on comprehensive and reliable building-specific data stands as a viable solution to this problem.

- **Sensor modeling:** The modeling of a sensor's detection zone within a building is intricate due to the presence of surrounding obstacles. This complexity results from distortion caused by attenuation and obstruction of detection signals. Consequently, employing binary sensing models and other oversimplified representations introduces a disparity between the network coverage observed in simulated environments and real-world scenarios. This disparity holds significant implications, particularly in critical applications.

Contributions of the thesis

Considering the aforementioned limitations, this thesis aims to bridge the gap between theoretical approaches to WSN deployment problem in indoor environments and real-world scenarios. The primary contributions of this thesis are succinctly summarized as follows:

- We provided an overview of the WSN deployment problem, focusing on problem modeling as outlined in the literature, the associated objective functions, and existing target area models. Also, we performed a comprehensive literature review on related artificial intelligence solutions with a particular focus on metaheuristics.
- To address the target area modeling challenge, we have developed a module based on the Building Information Modeling (BIM), which is robust data source providing detailed building characteristics. This module comprises two primary components: the data extraction component and the deployment zone modeling component. The data extraction component acts as a parser and extracts the semantic building data describing the building layout and obstacles from the BIM database. Subsequently, the deployment zone modeling component utilizes this extracted data to generate a series of deployment zones suitable for sensor deployment. Moreover, it transforms the raw obstacle data into a format readily usable for the deployment approach.
- We formulated the WSN deployment problem as a multi-objective combinatorial problem and proposed two connectivity repair heuristics based on Dijkstra and Steiner tree algorithms. These heuristics aim to rectify the connectivity of unconnected WSN topologies generated during the optimization process at a reduced cost.

- Given complex nature of the WSN deployment problem, particularly in large smart building areas, we initially considered merging the coverage and cost objectives into a single objective to simplify it. This was achieved by aggregating these objectives into a single fitness function. Subsequently, we conducted a comprehensive comparative study involving seven well-known metaheuristic algorithms frequently used in the literature for solving the deployment problem. Based on the obtained results, we proposed a mono-objective deployment strategy based on the Grey Wolf Optimizer. This approach incorporates a Steiner Tree-based heuristic to optimize the generated network typologies by minimizing the number of deployed sensors while preserving network coverage and connectivity.

- As coverage and cost present conflicting objectives, mono-objective approaches may have limitations in addressing the deployment problem. Hence, we developed a hybrid multi-objective metaheuristic to generate a set of Pareto-optimal connected sensor topologies with a good trade-off between coverage and cost. To design this approach, we conducted a comparative study involving four multi-objective algorithms tested on ten indoor architectures of varying sizes. The proposed method incorporates a two-stage local search to further enhance network coverage and reduce deployment costs of generated sensor networks.

- To enhance the precision of the deployment approach, we have designed a novel sensing model based on fully convolutional networks. This model accounts for the layout of the deployment environment and the presence of heterogeneous building obstacles, and their influence on the detection signals of electromagnetic sensors. The output of our sensing model is a signal strength map that identifies the zones covered by the given sensor within its surrounding environment.

Organization of the thesis

This thesis comprises two parts and 8 chapters. The first part covers the state of the art in WSN deployment across the first 3 chapters. The second part details the proposed approach for addressing the WSN deployment problem in smart buildings, presented in chapters 4 to 7. The content of each chapter is as follows:

- **Chapter 1** introduces sensor nodes, discusses their types and classification, and explores the application of WSNs in the context of smart buildings.

- **Chapter 2** addresses key aspects of the WSN deployment problem as presented in the literature, including problem definition, objective function modeling, sensing and communication models, and target area modeling.

- **Chapter 3** reviews artificial intelligence approaches for WSN deployment, with a focus on metaheuristics, hybrid metaheuristics, and machine learning solutions.

- **Chapter 4** discusses problem modeling, starting with a detailed BIM-based target area modeling, followed by problem formulation. It concludes by presenting our two proposed heuristics for addressing the connectivity constraint.

- **Chapter 5** presents a comparative study on WSN deployment using mono-objective metaheuristics. Subsequently, it introduces a hybrid approach that integrates the binary grey wolf optimizer with a Steiner tree-based heuristic to optimize deployment costs.

- **Chapter 6** examines multi-objective optimization strategies for solving the WSN deployment problem. It introduces a hybrid method that implements a two-stage local search to maximize network coverage and reduce deployment cost.

- **Chapter 7** introduces a detection zone estimation method for electromagnetic sensors in smart buildings using the U-net architecture with stacked dilated convolutions (SDU-net). The chapter also details the generation process of a dataset, encompassing both simple and complex indoor architectures.

- **Chapter 8** concludes the manuscript and gives future perspectives.

List of publications

Journal papers

- Khaoula Zaimen, Mohamed-el-Amine Brahmia, Laurent Moalic, Abdelhafid Abouaissa, Lhassane Idoumghar, "A Survey of Artificial Intelligence Based WSNs Deployment Techniques and Related Objectives Modeling," in IEEE Access, vol. 10, pp. 113294-113329, 2022, doi: 10.1109/ACCESS.2022.3217200 (IF 3.745, Rank Q1).
- Khaoula Zaimen, Laurent Moalic, Mohamed-el-Amine Brahmia, Abdelhafid Abouaissa, Lhassane Idoumghar, (2024). Multi-objective Approach for WSN Deployment in Smart Buildings based on BIM Database. Engineering Applications of Artificial Intelligence (submitted) (link: here).

International Conference Proceedings

- Khaoula Zaimen, Mohamed-el-Amine Brahmia, Laurent Moalic, Abdelhafid Abouaissa, Lhassane Idoumghar, "A Comparative Study of Meta-Heuristic Algorithms for WSN Deployment Problem in Indoor Environments," 2023 IEEE Congress on Evolutionary Computation (CEC), Chicago, USA, 2023, pp. 1-8, doi:10.1109/CEC53210.2023.10254142 (Rank A - ERA).
- Khaoula Zaimen, Laurent Moalic, Mohamed-el-Amine Brahmia, Abdelhafid Abouaissa, Lhassane Idoumghar, "Connectivity Repair Heuristics for Stationary Wireless Sensor Networks," ICC 2023 - IEEE International Conference on Communications, Rome, Italy, 2023, pp. 4804-4809, doi: 10.1109/ICC45041.2023.10278851 (Rank B - ERA).
- Khaoula Zaimen, Laurent Moalic, Mohamed-el-Amine Brahmia, Abdelhafid Abouaissa, Lhassane Idoumghar, (2024). A Hybrid Binary Grey Wolf Optimiser for WSN Deployment in Indoor Environments Based on BIM database. IEEE Wireless Communications and Networking Conference (WCNC) (accepted) (21-24 April 2024 in Dubai, United Arab Emirates, program link here).
- Khaoula Zaimen, Mohamed-el-Amine Brahmia, Laurent Moalic, Abdelhafid Abouaissa, Lhassane Idoumghar, (2021). Coverage maximization in WSN deployment using particle swarm optimization with Voronoi diagram. Advances in Model and Data Engineering in the Digitalization Era. MEDI 2021. Communications in Computer and Information Science, vol 1481. Springer, Cham. doi.

– Khaoula Zaimen, Mohamed-el-Amine Brahmia, Laurent Moalic, Abdelhafid Abouaissa, Lhassane Idoumghar, (2020). An overview on WSN deployment and a novel conceptual BIM-based approach in smart buildings. 2020 In 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS).

GitHub links for our projects code

– Code of the developed heuristics for repairing WSNs Connectivity: "Connectivity Repair Heuristics for Stationary Wireless Sensor Networks" (conference IEEE ICC'2023): GitHub link

– Code of the comparative study of mono-objective metaheuristic algorithms for deploying WSNs in indoor environments: "A comparative study of meta-heuristic algorithms for WSN deployment problem in indoor environments" (conference IEEE CEC'2023): GitHub link

– Code of the mono-objective approach based on Grey Wolf Optimizer for WSNs deployment: "A Hybrid Binary Grey Wolf Optimiser for WSN Deployment in Indoor Environments Based on BIM database" (conference IEEE WCNC' 2024): GitHub link

– Code of the comparative study of multi-objective algorithms, including the developed hybrid multi-objective algorithm with the proposed BIM-based target area modeling: GitHub link

– Code for the developed dataset for electromagnetic sensors in smart buildings: GitHub link

– Code of the proposed sensing model for electromagnetic sensors: GitHub link

Part I

WSN Deployment: State of the Art

CHAPTER 1

BACKGROUND ON WSNS AND SMART BUILDINGS

Wireless sensor networks play a pivotal role as foundational elements facilitating the advancement of intelligent environments and the development of smart urban structures. The incorporation of sensor networks, in conjunction with additional technologies and decision systems, within the framework of smart building structures has resulted in the transformation of these entities into comfortable living environments characterized by responsiveness, adaptability, and energy efficiency. In this chapter, preliminary concepts related to sensor networks are discussed first, followed by a brief presentation of smart buildings and their features. Finally, the applications of sensors within smart buildings are presented.

1.1 Preliminaries

1.1.1 Sensors

A sensor node is a tiny device that typically comprises four units [18]: sensing unit, processing unit, communication unit, and power unit (see Fig. 1.1), yet additional components, such as mobilizer and location finding system, may be added to fulfill other tasks. The sensing unit contains a sensor and an analog-to-digital converter (ADC). Thus, sensors detect events that occur within their sensing range, also called sensor detection zone, and then convert analog signals into digital signals for further analysis in the processing unit. Based on the sensor sub-unit, we distinguish two types of sensor nodes: contact and non-contact sensor nodes.

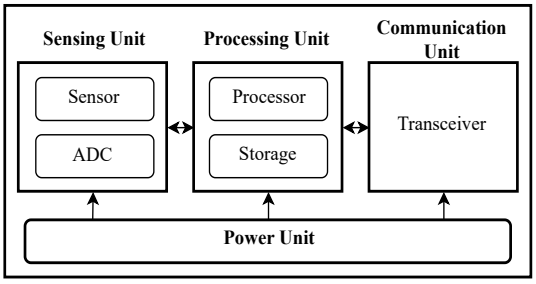


Figure 1.1: Sensor node architecture.

Contact sensor nodes require physical contact with the target in order to perform their measurements; examples of contact sensors include thermocouples, thermistors, and resistance temperature detectors. This type of sensors does not have a detection zone, but their readings are related to their local positions. In contrast, non-contact sensors, such as those utilizing the Hall effect and the Magnetoresistive effect, rely on physical phenomena that do not necessitate direct contact with the target. As a result, they can monitor wider areas. The second component is the processing unit, which consists of a processor for executing programs and a tiny storage unit for storing gathered data. The communication unit connects the sensor node to the network by enabling data transmission and reception. The last component is the power unit which supplies power to all operating parts of the sensor node.

Sensor nodes are classified based on three principal factors: the embedded sensing technology, the sensing type, and the sensing direction. Fig. 1.2 depicts the sensor sub-classes regarding each factor.

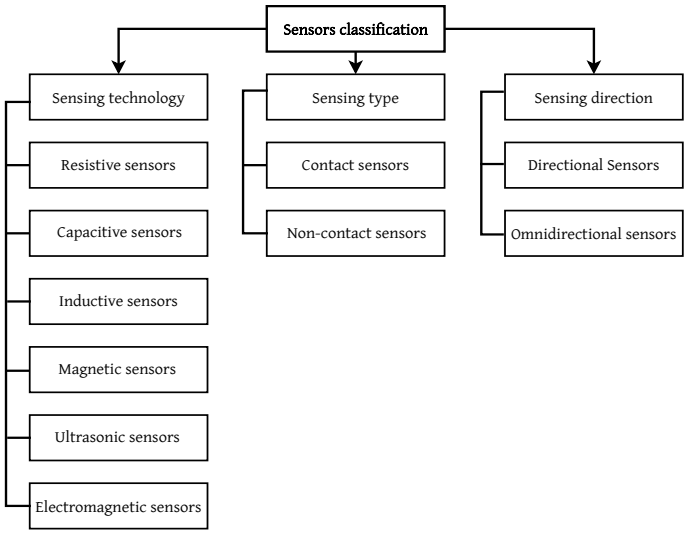
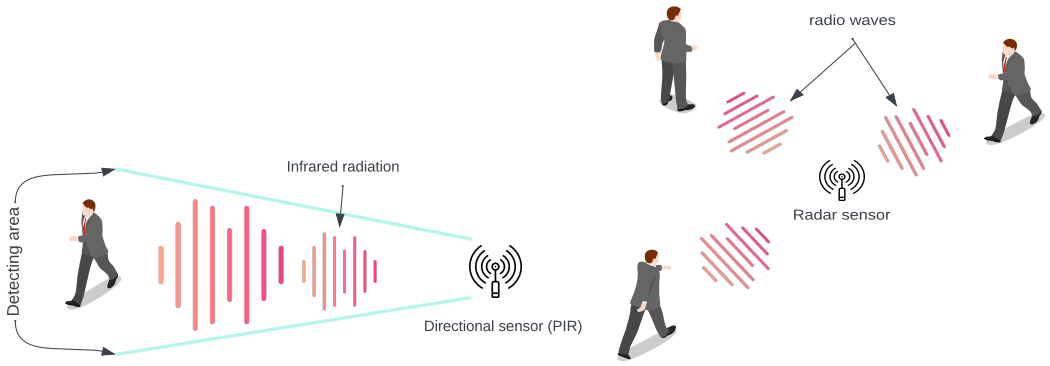


Figure 1.2: Classification of sensor nodes.



(a) Directional sensor (b) Omnidirectional sensor

Figure 1.3: Directional sensor vs Omnidirectional sensor.

Sensing technology defines the sensor behavior in response to a physical, chemical, or biological stimulus. This behavior can be translated into a variation of its electrical resistance, capacitance, inductance, etc., and then mapped into a suitable output value [19]. In addition, the sensor behavior is also affected by the sensing direction. A directional sensor node can perceive in only one direction at a fixed view [20]. In contrast, an omnidirectional sensor node has the ability to sense in all directions [20]. Fig. 1.3 illustrates an example of each type.

1.1.2 Wireless Sensor Networks

According to [21], a Wireless Sensor Network (WSN) is an infrastructure-less wireless network, consisting of multiple sensor nodes. Its principal function involves environmental monitoring, encompassing the collection of data on various physical parameters such as temperature, humidity, pollution, etc., as well as event detection, including gas leak and forest fire detection. A WSN can adopt various topologies based on the application requirements. The network topologies for a WSN are as follows:

- **Star topology:** In this communication topology, all sensor nodes send and receive messages to a single base station or sink node, with no direct interaction between individual sensors. The primary advantage of this topology is its simplicity in terms of setup and management. However, its suitability is constrained for networks deployed across expansive target areas due to the requisite placement of the base station within the communication range of each sensor.
- **Mesh topology:** In the mesh topology, sensor nodes can communicate with each other using either single-hop communication, which occurs within their direct communica-

tion range, or multi-hop communication, where messages are relayed through intermediate sensors. In multi-hop communication, two communicating sensors need to forward their messages through intermediate sensors. The principal advantage of this communication topology lies in its scalability and resilience to node failure, because of the redundancy of communication paths among sensors. However, it suffers from a higher energy consumption because of the use of multi-hop communication.

- **Tree topology:** In this communication topology, sensors are arranged hierarchically in a tree structure. Each node is permitted to communicate only with its preceding and succeeding nodes. This makes it susceptible to node failure as there exists only a single path between source and destination nodes.

- **Hybrid topology:** Hybrid topologies combine various network structures to leverage their respective advantages. However, their design and management could be a very complicated task.

1.2 Smart Buildings

The concept of smart buildings has diverse definitions in both industrial and academic sectors. Until now, there is no universally agreed-upon definition for the concept of smart buildings [22] [23]. Numerous definitions of smart buildings have been advanced since the 1980s [22]. Contemporary characterizations consistently assert that these structures exhibit high energy efficiency and possess the capability to discern and address occupants' requirements. Furthermore, they incorporate Building Energy Management Systems (BEMS) for the purposes of monitoring, control, and supervision, while also responding to external conditions such as climate [23]. The main features of smart buildings are summarized as follows [23]:

- **Real-time monitoring:** This entails the continuous gathering, analysis, and storage of data concerning physical parameters such as temperature, lighting control, and air pollution monitoring, along with data related to occupant tracking, energy consumption, water usage, etc.

- **Real-time interaction:** This entails instantaneous interaction between occupants and building systems, achieved through the adjustment of various services and control systems, including lighting and temperature. Additionally, it can be characterized by personalized settings that adjust the environment based on occupant preferences.

– **Energy management:** One of the most critical features of smart buildings is the reduction of total energy consumption. This is achieved through the incorporation of energy-efficient design implementations, including insulation and the integration of efficient Heating, Ventilation, and Air Conditioning (HVAC) systems customized to the present occupants' needs. Additionally, the integration of Renewable Energy Sources (RES), such as solar photovoltaic panels and wind turbines, enables the building to be energy-independent and less dependent on traditional grid power.

– **flexibility:** Flexible energy systems have been developed to address the challenge of efficiently managing building energy supply following demand throughout the day. The concept of 'Energy Flexible Buildings' was introduced by the International Energy Agency (IEA) and defined as the capacity of a building to manage its demand and production of renewable energy based on local climate conditions, user needs, and grid requirements. The ability of a building to provide energy flexibility is influenced by factors such as physical characteristics, technologies, control systems, and user behavior.

1.3 Types of sensors in smart buildings

In a smart building, various types of sensors can coexist, each representing a dedicated sensing system for a specific application. These sensors cover a wide range of functionalities, including monitoring environmental conditions such as temperature, humidity, and air quality, as well as controlling lighting, occupancy detection, activity recognition, and security systems [24]. The commonly used sensors in smart buildings are:

– **Temperature sensors:** Temperature sensors are widely applied in smart buildings to monitor temperature changes in the environment. Within these buildings, they play a crucial role in HVAC systems, where their primary function is to optimize energy usage. By providing accurate temperature readings, these sensors help regulate indoor temperatures efficiently at reduced costs. The most common types of temperature sensors are thermocouples and thermistors.

– **Humidity sensors:** Humidity sensors measure the level of water vapor present in the air within smart buildings. Their primary function is to identify issues such as dryness or excessive moisture within enclosed spaces in order to ensure a comfortable and healthy environment for building occupants.

- **Light sensors:** Light sensors, also known as photodetectors, are devices designed to detect light energy (photons) and convert it into electrical signals. Similar to temperature sensors, light sensors play a crucial role in managing building energy. They are commonly utilized in lighting control systems and light level monitoring applications, enabling more efficient use of energy resources within the building environment.

- **Air quality sensors:** These sensors are used in smart buildings to detect the concentration of particle pollution and measure the level of some harmful gases such as Carbon Dioxide (CO2) and Ozone (O3) in the indoor air. The reported measurements from these sensors are used to dynamically adjust HVAC systems to ensure optimal ventilation and air quality management throughout the building.

- **Smoke and fire detection sensors:** This type of sensor is used in smart buildings to enhance safety and security for building occupants. If smoke or fire is detected, these sensors promptly send alarms to enable quick evacuation of people and to implement intervention measures in order to mitigate any potential risks.

- **Occupancy sensors:** These sensors are utilized to detect the presence of people in a specific area. The primary sensing technologies for this type of sensors include infrared, ultrasonic, and microwave sensors. Their working principle is based on detecting changes in infrared radiation, ultrasonic waves, microwave signals, or other environmental factors caused by human presence, such as emitting CO2. Occupancy sensors are a cornerstone in various smart building applications that aim to optimize energy consumption and enhance the comfort of people. Common examples of these applications include dynamic heating and air conditioning adjustment, lighting control systems, as well as security systems.

- **Motion sensors:** Motion sensors play a pivotal role in smart buildings, serving various purposes ranging from enhancing security systems to optimizing occupant comfort. They are particularly crucial in applications requiring activity recognition, such as eldercare systems designed to detect falls. Motion sensors are primarily electromagnetic sensors that emit radio signals or microwaves. These sensors are based on radar technology, which involves analyzing the characteristics of reflected signals to identify occupant behavior within smart buildings. Their detection range is notably larger compared to other sensors; however, they are also more expensive. Therefore, deployment approaches considering the placement of these sensors should prioritize optimizing deployment costs.

Since contact sensors cannot cover large areas and generate measurements based on specific positions, they are strategically placed within smart buildings in strategic positions. For instance, temperature sensors, classified as contact sensors, require placement in specific areas far from heat sources and direct sunlight to ensure accurate temperature readings. The same applies to air pollution sensors, which are typically positioned in precise locations such as occupied spaces like meeting rooms or near sources of pollution susceptibility. Contact sensors should be positioned in unobstructed locations within smart buildings to ensure reliable monitoring over time and to prevent misreadings caused by blockage of airflow or other stimuli. Therefore, deployment strategies for sensors in smart buildings primarily focus on non-contact sensors mainly electromagnetic sensors. These sensors have the capability to monitor their surrounding areas effectively. However, their efficacy may be impeded in the presence of building obstacles, resulting in a diminution of their detection zones.

1.4 Applications of WSNs in Smart Buildings

WSNs play a crucial role in various applications within smart buildings and contribute to improving their energy efficiency and occupant comfort. Here are some examples of WSN applications in smart buildings:

- **Environmental Monitoring:** Environmental monitoring stands out as a primary application of WSN in smart buildings. It entails the collection of environmental and physical parameters, including temperature, indoor air quality, lighting, humidity, and more. This data is subsequently utilized and analyzed by intelligent building systems to dynamically adjust the indoor environment to ensure occupant comfort.
- **Occupancy Sensing:** Occupancy sensing involves identifying the presence and motion of individuals. These applications find utility also in activity recognition, particularly in the health and medical domains, where they are employed for tracking elderly or sick individuals in their homes.
- **Lighting Control:** To enhance occupant comfort and optimize energy consumption, smart buildings automatically adjust artificial lighting based on occupants' presence and natural lighting conditions, creating a comfortable lighting environment.
- **Safety Monitoring:** Ensuring occupant safety stands as a primary function of smart buildings. This is achieved through the deployment of various safety sensors, including

those for fire and smoke detection, gas, and water leaks. The data collected by these sensors is then utilized by safety systems to initiate early actions and minimize potential damage.

1.5 Conclusion

In this chapter, we introduced sensor nodes and WSNs, provided a definition of smart buildings and their distinctive features, and illustrated examples of sensor network applications in such environments. In the subsequent chapter, we will define the WSN deployment problem. We will present the related objectives modeling that exist in the literature as well as the sensing and communication models describing sensor signal propagation. Finally, we will discuss diverse modeling approaches used in literature to characterize target areas for sensor network deployment.

CHAPTER 2

WSN DEPLOYMENT METRICS AND TARGET AREA MODELING

2.1 Introduction

WSN design and deployment is a complex task since it has a direct impact on the performance of the network and consequently on the applications using it. Furthermore, several critical applications such as health care, military, or even environment monitoring applications require a specific degree of quality of service (QoS), namely coverage, cost, connectivity, and network lifetime. Therefore, the research community has suggested several metrics also called optimization objectives, to tackle the problem of WSN deployment. Each objective may have various mathematical models, each with a certain accuracy level. In this chapter, we will present a general reference optimization model for the WSN deployment problem, including the decision variables, the salient optimization objectives, and the feasible constraints considered in the WSN deployment problem.

2.2 WSN deployment problem

Solving the problem of WSN deployment consists of finding the best positions for sensors in the target area to ensure optimal coverage, connectivity, and other performance metrics. This entails considering several factors regarding the sensor characteristics and the target area, as well as its corresponding obstacles. The decision variables of the WSN deployment method refer to the locations of sensor nodes in the target area. There are

mainly two representations of the decision variables, the vector representation, and the grid representation. In the former representation, the deployment scheme is defined as an array of Cartesian coordinates where the cell i represents the position (x_i, y_i) of the sensor i. Each decision variable (x_i, y_i) must fulfill the constraints of the upper and lower bounds of the area of interest. In the latter representation, the deployment solution is defined by a grid of L rows and W columns. Each cell $cell_{i,j}$ of the grid corresponds to specific zone in m^2 in the real environment. Also, each value $V_{i,j}$ of the $cell_{i,j}$ represents a binary decision variable, defined as follows: $V_{i,j} = 1$, if the cell contains a sensor node, whereas $V_{i,j} = 0$ otherwise. The grid representation could be flattened and converted into a binary vector.

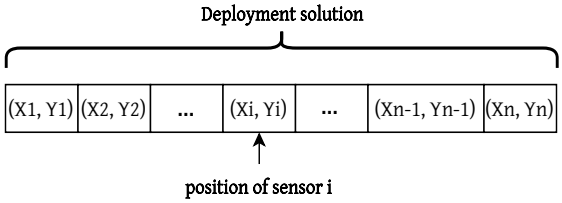


Figure 2.1: The vector representation of the decision variables

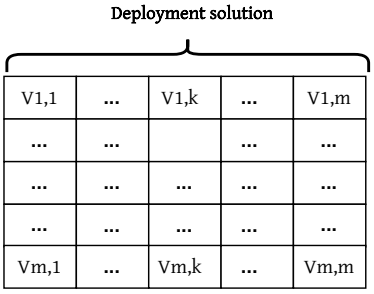


Figure 2.2: The grid representation of the decision variables

2.3 Objective functions modeling

In this section, we will review the most important metrics considered during the deployment of sensors in both indoor and outdoor environments. These metrics include network coverage, deployment cost, network lifespan, and energy consumption. In the deployment process, these metrics are regarded as objective functions that must be optimized. Thus, we will provide in this section their corresponding modeling, as outlined in existing literature.

2.3.1 Coverage

The primary function of WSNs is to monitor the environment and sense specific events. Therefore, coverage is identified as a salient performance metric that must be prioritized in WSNs deployment design. Mainly, coverage is classified into three types: target (point) coverage, barrier coverage, and area coverage.

Target coverage: in target coverage, sensor nodes are deployed to monitor a set of target points that could be static [25], or mobile [26]. This type of coverage is widely used in military applications in which a set of locations must be controlled. Another variant of the target coverage problem is Q-coverage [27]. This latter adds QoS requirements, such as each target point should be covered by a predefined number of sensors. Moreover, a periodic target coverage called sweep coverage was tackled in the literature [28, 29, 30]; it seeks to deploy fewer mobile sensors to monitor a set of target points. The real challenge with this coverage type is scheduling a small number of mobile sensors to periodically supervise target points while consuming as little energy as possible.

Barrier coverage: It is used to protect the borders of critical regions or infrastructures such as territory frontiers from intruders that try to penetrate them [31]. A strong barrier coverage is provided by deploying sensor nodes in irregular belt shape to form a barrier with no gaps so that intruders can not traverse the region whatever the crossing path they use.

Area coverage: The goal of area coverage is to monitor a target region so that every location is within the sensing range of one or more sensor nodes[32]. Classical area coverage methods often assume a 1-coverage, meaning that every point in the region of interest must be covered by at least one sensor node. Yet, critical applications such as gas leakage explosions require higher accuracy. For that, researchers use the k-coverage technique [33] to ensure that every location in the target area is within the sensing ranges of k sensor nodes.

The coverage function depends mainly on the sensing model and the environment modeling:

– Impact of the sensing model: According to [34], the sensing model is the mathematical formula used to estimate the probability that a target point is within the detection zone of a sensor node. Thus, the coverage function applies it to assess the detection zone for each sensor in the network. Consequently, any inaccuracy of the sensing model will result in erroneous estimation of the full network coverage since it could lead to a significant disparity between real and predicted sensed data and skew any information derived from this data. This will be a serious network performance issue, especially for mission-critical applications requiring high QoS. The sensing models used in the

literature are detailed in Section 2.5.1.

– Impact of the environment modeling: Accurate target area modeling enables measuring the detection zone of each sensor node in relation to the surrounding obstacles, thereby improving coverage estimation of the coverage function. A basic environment modeling, on the other hand, imposes strong assumptions that may not be met in reality, and so the coverage estimation may not reflect the true coverage of the sensor network. This will be more detailed in Section 2.6.

There are primarily two approaches for assessing the network's overall coverage in the case of coverage area: the grid model and the sensing zones aggregation model. The matrix technique depicts the region of interest as a grid, with sensors positioned in the center of cells. Each cell is meant to be covered if its center is within the detection zone of a sensor node. In this case, the coverage model will be the ratio of covered cells to the total number of cells:

$$Coverage(Z,S) = \frac{\sum_{i=1}^{H\times W} Cov(cell_i,S)}{H \times W} \tag{2.1}$$

Where H and W represent the length and the width of the matrix (area of interest) respectively, S is the set of sensor nodes to be deployed, Z is the whole zone (all the cells in the matrix), and $Cov(cell_i,S)$ is the probability that the $cell_i$ is covered by the set S and it is computed using a sensing model. If the cells of the grid have different degrees of importance, then each coverage probability of a given cell will be multiplied by a preset weight. The sensing zones aggregation model considers the union of all sensor detection zones. This method is more accurate than the grid method since it computes the actual covered regions. It is computed as follows :

$$Cov(Z,S) = \frac{\| \cup_{j=1}^{\|S\|} detZ(s_j)\|}{Z} \tag{2.2}$$

Here the $detZ$ function computes the sensing zone of a given sensor.

2.3.2 Network lifetime

WSN lifetime represents the duration in which the network can fulfill its mission properly. It has several definitions that coexist in the literature. It can be described as the time duration of the sensor network until the first sensor node runs out of energy [6]. Its

mathematical model can be formulated as follows:

$$Lifetime_{network} = \min(lifetime(node_i)_{i=1,..,N}) \tag{2.3}$$

The network lifetime is also defined as the ratio of the time until one of the sensor nodes run out of energy, i.e. the time until the first sensor node failure to the maximum lifetime of the sensor network. It is also modeled as follows:

$$Lifetime_{network} = \frac{\min\{T_{failure_i}\}_{i=1,..,N}}{T_{max}} \tag{2.4}$$

Where $\min(T_{failure_i})_{i=1,..,N}$ represents the maximum number of sensing cycles before the first sensor node runs out of energy and T_{max} is the maximum sensing cycles of the network.

2.3.3 Energy consumption

Energy consumption is a crucial concern in WSNs deployed in challenging environments since sensor nodes are energy-constrained devices. They consume energy while sensing the environment, processing, transmitting, and receiving data. Moreover, in some situations, the communication subsystem could engender other sources of wasted energy [35], such as in the case of packet collisions, overhearing, control packet overhead, idle listening, and interference [35]. Therefore, various solutions for energy conservation were proposed in the literature to extend the WSN lifespan [36, 37, 38, 39, 40]. Each solution deals with a specific aspect such as data reduction-based techniques, duty cycling techniques, and energy-efficient routing. We distinguish two major approaches in data reduction-based techniques: data prediction and data compression. The data prediction approach attempts to describe the sensed data by establishing a model. This latter will be exploited to generate data instead of using real gathered ones, and it can be built using stochastic approaches, time series forecasting, and algorithmic approaches [41]. On the contrary, the data compression approach uses the real sensed data while decreasing the number of bits that must be transferred; hence, the energy used for communication will be preserved. Duty cycling techniques schedule the activity of sensor nodes [35] so that sensors are switched off when they do not impact the network's functionality. Another technique used for energy conservation is the design of energy-efficient routing protocols that intend to find the most effective path for end-to-end packets transmission while considering the residual energy for each sensor node. The mathematical model [42] used to

describe the energy E_{cons} consumed by a set of sensor nodes in a given path is:

$$E_{cons} = \sum_{k=1}^{N}((t_k^{access} + t_k^{process}) * E_k^{operate} + E_k^{trans} * t^{msg}) \tag{2.5}$$

Where t_k^{access} and $t_k^{process}$ correspond to the time needed by node k to acquire and process data respectively, N represents the number of nodes in the path, t^{msg} is the message transmission time duration, $E_k^{operate}$ and E_k^{trans} are the operational power and transmission power of node.

2.3.4 Deployment cost

The deployment cost is an essential factor in designing the WSN deployment scheme. It can be defined as the total cost of purchasing and positioning all sensor nodes in the target area. It can be represented as follows:

$$Cost_{wsn} = cost_{sensor} * |SN| \tag{2.6}$$

Where SN represents the number of sensor nodes and $cost_{sensor}$ represents the price of purchasing and positioning one sensor.

2.4 Feasibility constraint: Network connectivity

Multiple constraints may be taken into account during the deployment process, depending on the application requirements. However, it is imperative that network connectivity is considered in all deployment methodologies to ensure the generation of feasible network topologies. Without complete connectivity among sensor nodes, there is a risk of losing gathered data which my lead to inaccurate monitoring. Two sensor nodes are connected if they can exchange data in both directions. A sensor network is connected if there exists at least one path between each pair of sensors. Furthermore, a k-connectivity with ($k >= 1$) means that there exist at least k distinct communication paths between each pair of sensors. Several models have been proposed in the literature to assess the communication probability between two sensors as highlighted in Section 2.5.2.
The connectivity constraint of WSN in the deployment problem has been handled in the literature following one of these techniques:

- **Imposing penalties on unconnected topologies:** This technique involves adding a penalty to the fitness function of unconnected topologies to reduce their selection probability. The primary drawback of this technique lies in the absence of a guarantee to produce connected topologies even in the case of big population sizes. Furthermore, the likelihood of the best topology being unconnected increases significantly when dealing with extensive sensor networks deployed in large target areas with obstacles.

- **Adapting the operators of optimization methods to exclusively generate connected topologies:** This technique is based on the adjustment of the optimization operators or position update techniques used by optimization algorithms to generate new solutions under the connectivity constraint. The main drawback of this technique lies in its complex and challenging implementation and limited applicability across various optimization algorithms. Additionally, it restrains the exploration capability of the optimization algorithm by restricting the examination of potential search regions. Consequently, this limitation makes the algorithm converge towards potential local optima, thereby decreasing population diversity.

- **Repairing connectivity of unconnected topologies during each iteration of the optimization process:** This approach addresses the limitations of the previously mentioned techniques. It retains the same optimization operators and position update techniques to allow a broader exploration of the search space. After each optimization iteration, unconnected topologies are repaired by adding the minimum number of necessary nodes to establish connectivity between sensor sets.

2.5 Sensing and communication models

2.5.1 Sensing models

Several sensing models have been proposed in the literature to estimate the detection zone of sensor nodes [43, 44, 45, 46]. These models are categorized into omnidirectional and directional sensing models, depending on the direction of the sensing range [20]. Most of these models do not consider the environmental impact (shadowing and signal attenuation) and sensor characteristics simultaneously. This issue affects the computation of sensors locations since the models, in most cases, do not reflect real-world scenarios. In what follows, we will define the most frequently used sensing models, which we have

divided into two categories: deterministic models and probabilistic models. We compare between this models in Table 2.1.

Deterministic sensing model

Also called the Boolean model or the Binary model, is the most commonly used model in the literature because of its simplicity. This model assumes that the detection zone of a sensor node is a uniform disk of radius R_s (R_s is the sensing range of the sensor). That is to say, any event that occurs within the disk will be captured by the sensors; otherwise, it can not be detected. This model considers only the Euclidean distance between the sensor node and events or target points and does not consider other external factors such as obstacles or signal strength. The probability detection of this model is given in Eq.2.7:

$$P_{det}(P,s) = \begin{cases} 1 & \text{if } d(P,s) \leq R_s \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Where $d(P,s)$ refers to the Euclidean distance between the point P and sensor s, R_s is the sensing range of sensor s, and $P_{det}(P,s)$ is the probability that the target point or the event P is within the sensing range of sensor s.

Probabilistic sensing model

It assumes that a probabilistic distribution models the sensing range of a sensor and any event that occurs within the sensing zone will be detected with a certain probability. This latter depends on the sensing model applied. In what follows, we will present the principal probabilistic models found in the literature.

- Sigmoid model: This model was used in [47] to conceive membership functions for sensing range (distance) and sensing angles, and then the final detection probability is the multiplication of membership functions. The probabilistic membership function of distance is given as follows:

  $$P_{det}(d) = 1 - \frac{1}{1 + e^{\beta(d - R_s) - t_d}} \tag{2.8}$$

  Where d is the Euclidean distance between the event or the target point and the sensor, β and t_s are two adjustable parameters according to the characteristics of the sensor, and R_s is the sensor sensing range.

– Attenuated disk model: This sensing model assumes that the sensing ability of a sensor decreases when the distance separating it from the event (or target point) gets longer [48]:

$$P_{det}(d) = \frac{\lambda}{d^{\alpha}} \tag{2.9}$$

Where λ is a constant, d represents the Euclidean distance between the sensor and the target point, and α is the path attenuation exponent reliant on the environment.

– Probabilistic model with noise: this sensing model is similar to the attenuated disk model, yet it considers the impact of the environment on the sensing ability of the sensor. For this, it includes a noise energy η that follows the Gaussian distribution.

$$P_{det}(d) = \frac{\lambda}{d^{\alpha}} + \eta \tag{2.10}$$

– Exponential model: this model measures the sensing attenuation based on the distance d between the sensor and the target point [49].

$$P_{det}(d) = e^{-\alpha d^{\beta}} \tag{2.11}$$

α and β represent the degree of sensing attenuation.

– Shadow fading model: in this model, the sensing ability of a sensor node is not regular in all directions because of the existence of obstacles [50].

$$P_{det}(d) = Q(\frac{10\eta log_{10}(d/r_s)}{\sigma}) \tag{2.12}$$

where,

$$Q(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-y^2/2} dy$$

η is the path loss exponent, σ is the shadowing parameter, and d and r_s are respectively the distance between the sensor and the event and the sensing radius.

– Elfes sensing model: This model considers both the distances between the sensor

and the event and the sensor's physical properties. It is defined as follows [50]:

$$P_{det}(d) = \begin{cases} 1, & d \leq R_1 \\ e^{\lambda(d-R_1)^\gamma}, & R_1 < d < R_{max} \\ 0, & d \geq R_{max} \end{cases} \tag{2.13}$$

Where R_{max} is the maximum sensing radius of the sensor, R_1 represents the certainty zone of the sensor detection, λ and γ are fixed based on the sensor's physical characteristics.

– Hybrid model: this model was proposed in [51], it combines the Elfes sensing model and Shadow fading model for the purpose of considering both the sensor characteristics and the environmental factors simultaneously.

$$P_{det}(d) = \begin{cases} Q(\frac{10nlog_{10}(d/r_s)}{\sigma}), & 0 \leq d \leq R_1 \\ \min(e^{\lambda(d-R_1)^\gamma}, & \\ \frac{10nlog_{10}(d/r_s)}{\sigma}), & R_1 < d < R_{max} \\ 0, & d \geq R_{max} \end{cases} \tag{2.14}$$

Here d represents the Euclidean distance between the target and the sensor node, and the remaining parameters are the same as in the Elfes and Shadow fading sensing models.

2.5.2 Communication models

Several mathematical models have been proposed in the literature to quantify the connectivity between sensor nodes. The commonly used model is the binary communication model (see Eq. 2.15), which considers that a sensor node can send data to another sensor if the Euclidean distance between the two nodes is less or equal to the minimum of their communication ranges.

$$P_{con}(s_i, s_j) = \begin{cases} 1 & \text{if } d(s_i, s_j) \leq R_c \\ 0 & \text{otherwise} \end{cases} \tag{2.15}$$

Table 2.1: Sensing models in the literature

Sensing model	**Parameters**	**Considered sensor physical features**	**Considered elements**	**Limits**
Deterministic model	None	Sensing range	Distance d	Unrealistic model. Does not consider sensor characteristics. Does not consider the impact of the environment (shadowing and signal attenuation). Overestimation of the sensing ability.
Sigmoid model	β, t	Sensing range Directional angles Other hardware characteristics	Distance d Reference angles	Does not consider the impact of the environment. Assumes a uniform sensing ability in all directions. Overestimation of the sensing ability.
Attenuated disk model	λ, α	None	Distance d	Assumes idealistic environment without obstacles. Does not consider sensor characteristics. Overestimation of the sensing ability.
Exponential model	α, β	None	Distance	Assumes idealistic environment. Does not consider sensor characteristics. Assumes a uniform sensing ability in all directions. Overestimation of the sensing ability.
Probabilistic model with noise	λ, α	None	Distance	Does not consider sensor characteristics.
Shadow fading model	η σ	None	Distance d	Does not consider sensor characteristics. The path loss is the same in all directions.
Elfes model	λ, R_{max}, R_1	Sensing range Other hardware characteristics	Distance d	Does not consider the impact of the environment (shadowing and signal attenuation).

Other reliable models based on signal propagation have been proposed to assess the connectivity between nodes mathematically. Radio propagation models aim to estimate the behavior of signal spreading in different environments [52]. Indeed, a signal may encounter several types of obstacles according to the environment it crosses, and therefore, it could be scattered, refracted, reflected, and diffracted [53]. According to [54], signal propagation modeling methods are mainly categorized into four types:

- Deterministic models: There are very high accuracy models that simulate the signal propagation in a specific location since they apply physical laws on 3D data describing the environment. These models are costly in terms of computing resources and time, and the commonly used models of this category are Ray-Tracing and Ray Launching.

- Stochastic models: These models use random variables to describe the randomness of the radio channels [54]. Hence, they are highly employed in large scale fading and small scale fading modeling. The Rayleigh fading model and Rice fading model are the most known ones in this category.

- Empirical models: These are the most used models in the field of network design because of their simplicity and low computational time. An empirical model is based on a huge collection of measurements related to a specific situation (system parameters, environment and type of communication system) [55] to predict the signal path loss.

– Semi-deterministic models: These are a combination of deterministic models and stochastic or empirical models [54]; thus, they are assumed to be more precise than stochastic or empirical models and consume lower computational resources than a deterministic model; an example of this category is the Dominant path model.

2.6 Target area modeling

The deployment scheme strongly depends on the target area characteristics, namely its form, dimensions, type (indoor/outdoor), and obstacles. The form and dimensions are used to outline the borders and define the potential deployment locations within the area of interest. The obstacles allow for a more refined selection of the possible deployment points by excluding the locations where they are present. Therefore, having a reliable data source that covers all the features of obstacles, precisely their thickness, materials, widths, and heights, is necessary for assessing their impact on the sensing and communication zones of the sensor nodes and hence obtaining a realistic deployment result. Indeed, the phase of target area modeling is often neglected in the proposed solutions as most of them suppose a 2D free obstacle layout. When considering obstacles, these are illustrated as dispersed regular or irregular polygons and could be homogeneous or heterogeneous:

– Homogeneous obstacles: Described as opaque objects [56, 57, 58, 59, 60, 61] that completely hinder the signal transmission. Hence, the deployment solution avoids positioning sensors in the vicinity of obstacles to maximize coverage even further.

– Heterogeneous obstacles: They attempt to simulate a real-life environment by incorporating many sorts of obstacles, each with a different attenuation value that defines how the signal intensity is affected [62, 63, 64].

2.6.1 Indoor vs Outdoor

Most of the reported WSNs deployment solutions do not specify the type of environment, whether indoor or outdoor and assume that the same deployment scheme can be adopted in both. However, the two environments have different properties, as explained below:

– Types of obstacles: One of the main differences that should be highlighted in the solution design of an indoor and an outdoor deployment is the types of obstacles existing in both environments. An indoor environment refers to all types of buildings (houses,

Table 2.2: Data sources used to model the target area in WSNs deployment problem

Data source	Environment type	Dimension of environment	Type of obstacles	Shape of obstacles	Position of obstacles
BIM (IFC file)	Indoor	3D	Included	Included	Included
Grid model of DEM	Outdoor	3D	Not included	Not included	Included
TIN model of DEM	Outdoor	3D	Not included	Not included	Included
Contour model of DEM	Outdoor	3D	Not included	Not included	Included
Satellite image	Outdoor	2D	Included	Included	Included

hospitals, malls, schools, etc.), and it is primarily characterized by the construction materials constituting walls and ceilings, woods, glass, etc. In contrast, an outdoor environment could refer to a city, a forest, a mountain or even an ocean, and it may contain different types of obstacles: trees, buildings, water, rocks, etc. Therefore, the WSN placement should be adjusted according to the target area obstacles in order to have better performance.

– Data source describing the target area: As mentioned before, having a data source describing the area of interest is very important to achieve a realistic deployment. A complete data source that can describe a building accurately and provide the necessary information (separators, plans, windows, materials, etc.) needed to model the target area is the Building Information Modeling (BIM) tool [65]. For outdoor environment, researchers use mainly the Digital Elevation Model (DEM) [66, 56, 67] and raster and vector modelings [68] which provides a 3D representation of the terrain topology. However, this data source remains incomplete since it does not contain all the information related to the target area, namely the terrain type. Table 2.2 summarizes the commonly used target area modeling used in literature.

2.6.2 2D vs 3D Deployment

The area dimension is another important criterion to be considered in the process of target area modeling. Indeed, most of the existing research works assume a 2D flat area divided according to a regular pattern as the grid representation [62, 64] or using a computational geometry approach such as Voronoi diagram and Delaunay triangulation [69, 70, 71]. Other approaches do not adopt any area division technique but define a set of distributed deployment points where sensors could be placed. Both representations cannot

describe real-world scenarios since they do not lead to a realistic assessment of the coverage, connectivity, and deployment cost. Thus, a more complex 3D modeling is required to simulate the real behavior of WSN [72], mainly the propagation of its communication and sensing signals. Indeed, 3D area modeling allows us to consider the elevation of obstacles and terrain at each point in the target area and thus, their impacts on the line of sight between target points and sensors as well as the obstruction degree of the sensing and communication zones of the sensors. The 3D WSN deployment has been proven to be more challenging and necessitates more sensor nodes to reach the same coverage rate as a 2D WSN deployment [73]. The 3D grid division [74, 75, 76] and Digital elevation model (DEM) [77, 56, 66] are the two widely used target area modelings in 3D environment.

2.6.3 Target area modeling for indoor environments

The topology of WSNs operating within indoor environments, particularly smart buildings, heavily relies on specific characteristics of the target area. These characteristics include the internal architecture, spatial dimensions, and building obstacles [5]. The majority of proposed methodologies addressing WSN deployment in indoor indoors tend to oversimplify the representation of the target area. This oversimplification results in a significant disparity between the real network performance affected by physical building obstacles, and the performance assessed through simulation. The existing models for indoor environments often adopt simplifications, such as the use of 2D grids devoid of obstacles [78], which represents the most basic form of modeling. Some progress has been made with models incorporating obstacles within 2D grids [64]. Additionally, other approaches have introduced 2D planes including opaque obstacles [79] or heterogeneous obstacles [62, 80]. More advanced models have extended their scope to encompass 3D representations, as seen in [60], where the deployment area is divided into multiple horizontal planes with varying heights and a vertical plane to simulate walls. Similarly, in [81], the deployment area is characterized by ceilings and walls positioned at a height of 2 meters above the floor. However, these established modeling methodologies often fall short, either by failing to consider all pertinent building-related data or requiring complex remodeling when considering various target areas. Consequently, the development of an automated solution capable of modeling any given target area based on comprehensive and reliable building-specific data stands as a viable solution to this problem. Such an approach would not only bridge the gap between simulation and real-world performance but also significantly enhance the credibility and accuracy of the results obtained from these simulations.

2.7 Conclusion

In this chapter, we have defined the WSN deployment problem and detailed its decision variables and feasibility constraints. Also, we presented mathematical modeling of the most salient objective functions, including the coverage function, network lifetime, energy consumption, and cost function, along with the sensing and communication models. Furthermore, we emphasized the importance of environment modeling in the design of the deployment scheme and recapitulated the commonly used target area modeling reported in the literature. In the next chapter, a variety of deployment solutions utilizing AI-based techniques, including evolutionary algorithms, swarm intelligence optimization algorithms, hybrid metaheuristics, and machine learning will be reviewed and analyzed.

CHAPTER 3

REVIEW ON WSNS DEPLOYMENT USING ARTIFICIAL INTELLIGENCE

3.1 Introduction

The WSN deployment problem is one of the real-world complex optimization challenges that involve optimizing multiple objectives under connectivity constraint. Due to the non-convex nature of the WSN deployment problem, exact methods, despite the progress in the field, remain efficient only for deployment problems in small-scale target areas [16]. Handling larger WSN deployment areas with exact methods could involve significant computational complexity due to their limited scalability. Consequently, the deployment approaches proposed in the literature are mainly based on Artificial Intelligence techniques, offering enhanced scalability and ability to address the non-convex characteristic of WSN deployment optimization.

Artificial intelligence (AI) is the branch of computer science that focuses on finding solutions to complex real-world problems requiring human intelligence. In the scope of the WSNs deployment problem, metaheuristics are the most used AI techniques by the research community to compute the relevant deployment scheme. Very few initiatives are based on machine learning namely the Q-learning to heal coverage holes by redeploying sensors. In this section, we survey WSNs deployment solutions based on various AI techniques namely evolutionary algorithms, swarm intelligence-based algorithms, hybrid algorithms, and machine learning algorithms.

3.2 Metaheuristics

Metaheuristic algorithms are AI techniques defined as general-purpose algorithms that can be applied to tackle a wide range optimization problems [17]. The effectiveness of these algorithms in generating solutions depends on factors like problem complexity and the exploration-exploitation mechanisms they employ. Depending on the number of objective functions targeted for optimization, metaheuristics can be characterized as either mono-objective, for single objectives, or multi-objective optimization algorithms. Moreover, they can be further categorized into subtypes such as evolutionary algorithms and swarm-based algorithms, based on their overall framework. In the subsequent sections, we delve into an examination of metaheuristic-driven approaches proposed for deploying WSNs in the literature.

3.2.1 Evolutionary algorithms

Evolutionary intelligence is a branch of bio-inspired algorithms that relies on population concept and biological heredity [82] which means transferring features from parents' generation to children's generation. The population concept allows for the simultaneous search for the optimal solution in more than one direction using individuals. An individual represents the encoding of a solution for a given optimization problem. The individuals of iteration i are called parents, and the individuals of iteration $i + 1$ are called children. Parents share the search information with children through evolutionary reproduction operators. Each individual has a score that denotes how well it solves a particular problem. Individuals with a high score will replace parents in the next population and cooperate in producing new individuals with evolved performances. There have been numerous applications of evolutionary algorithms in solving real-world problems [83], including the problem of WSN deployment. [62, 63, 84, 85, 86, 56, 64, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96]. In what follows, we will emphasize the most prominent evolutionary algorithms with their related WSNs deployment approaches.

Genetic algorithm (GA): It is one of the evolutionary algorithms that have been widely applied in diverse problem domains. Its pseudo-code is depicted in the Algorithm 1.

Algorithm 1: Pseudo code of the genetic algorithm

```
Initialize population;
Compute fitness for each chromosome;
while Termination condition is not satisfied do
    Selection of parents to generate offspring;
    Recombine parents (crossover);
    Mutate children;
    Compute fitness for new individuals and update population;
end
```

Numerous research works have used GA to solve the WSNs deployment problem. Each work tried to propose sophisticated operators and individual coding. In [62], authors developed an optimizer based on a constrained multi-objective genetic algorithm. They considered the weighted sum method to combine two objectives of maximizing coverage and minimizing cost under the limited budget, the coverage, and connectivity degrees constraints. Additionally, the authors proposed a new individual encoding to model the position of heterogeneous sensors within the area of interest and used the elitist selection and new mutation operator that allows removing, adding, and moving the sensor nodes. The same authors applied the GA and the weighted sum technique in [63] to tackle the problem of WSNs deployment in an indoor environment. They combined both coverage and connectivity according to predefined network topology. The deployment space is depicted as a matrix, and each chromosome's gene defines whether the associated cell of the matrix is occupied by a sensor node, a router, or empty. Authors in [84] conceived a multi-objective genetic algorithm to optimize the placement of sensor nodes. The proposed algorithm aims to widen the coverage range and reduce energy consumption by decreasing the number of active nodes. The authors used the binary coding that considers the active state of sensor nodes, i.e. a gene at position i is set to 0 if the sensor i is in sleeping mode; otherwise, it is set to 1, and the corresponding sensor is on working mode. Furthermore, the authors presented an enhanced fitness function that analyzes the number of potential additional covered target locations before deciding whether or not to activate a sensor node, and they compared the single-point and multi-point crossover operators. The simulation results show that the multi-point cross-over allows to achieve better results in terms of convergence rate and optimization objectives. Another work in [85] has applied the GA to tackle the problem of the heterogeneous sensor nodes deployment within a 2D area with obstacles. The proposed model focuses on area coverage optimization under the connectivity constraint. Moreover, it includes an improved population initialization based on a modified virtual force algorithm and a fitness function that

considers the coverage overlap between each pair of sensors and the overlaps between sensors and obstacles. In [97], the authors aimed to prolong the network lifetime using caching mechanism as a means to reduce data transmission and network latency. Thus, they applied the GA to compute the optimal positions of cache nodes that allow covering a maximum number of requesting nodes. Simulation results showed that the proposed solution achieved better performance in terms of average latency and the total number of messages compared with other existing methods. *Kosar and Ersoy* have tackled in [98] the problem of sink node placement in a 3D environment for border surveillance. The basic scheme of their approach is based on a Discrete WSN simulator and the GA-based optimizer. The former component is used to compute the network lifetime which represents the fitness function. It simulates the sensing and communication functions of the WSN in a given terrain elevation map while computing the network lifespan. In the optimizer component, an individual represents the location of the sink node.

Table 3.1: Main WSN deployment approaches based on Genetic Algorithm

Work	Year	Objective	Constraints	Dimension	Obstacles
[62]	2020	Coverage K-coverage Overlapping coverage Cost	M-connectivity Limited budget	2D	Heterogeneous
[63]	2020	Coverage K-coverage Overlapping coverage Cost	Connectivity	2D	Heterogeneous
[85]	2020	Coverage	-	2D	Homogeneous
[84]	2017	Coverage Energy consumption	Ensure at least p% coverage (p is input data)	2D	-

Non-dominated Sorting Genetic Algorithm (NSGA-II) Another evolutionary algorithm that has been widely applied in solving WSNs deployment problem [56, 64, 87, 88, 89, 90] is NSGA-II [99]. This metaheuristic is a multi-objective optimization algorithm with two main aspects: fast non-dominating sorting and crowded distance. Authors in [56], conceived a two objectives NSGA-II based approach with a guided crossover and mutation operations for WSNs deployment problem in a 3D environment. In their work, the authors suggested an individual coding scheme that incorporates information on sensor nodes' locations and directions, and they computed coverage using a probabilistic sensing model presented in [47] with an improved visibility function based on the Bresenham line-of-sight algorithm. *Benatia et al.* [64], sought to find the near-optimal solution for WSNs deployment in smart buildings through two evolutionary algorithms: GA and

NSAG-II. Therefore, the proposed approach deals with multiple objectives: deployment cost, coverage, connectivity, and over-coverage. According to the authors, the choice of the adequate metaheuristic algorithm depends on the user requirements; the NSGA-II is recommended when the approach is not a-priori. *Dahmane et al*. [87], dealt with the problem of deploying temperature sensor nodes in smart buildings; their approach is based on NSGA-II with two objectives to optimize coverage and cost. In their solution, the authors included BIM database information to model obstacles of the building. Indeed, their coverage model depends mainly on the distance between the target point and the sensor node and the heat flow resistance of materials constituting obstacles. *Khalesian and Delavar* proposed in [88] a constrained Pareto-based multi-objective evolutionary approach that attempts to reach a trade-off between the network coverage and the energy consumption objectives while maintaining the sensors connectivity. They modeled the sensors network as a connected graph of k sensor nodes and $k-1$ edges referring to the communication links between nodes. Then, they conceived two crossover operators that allow for generating feasible solutions. The first operator combines two parent graphs to form a graph with $2k$ nodes and $2k-2$ edges; next, it randomly selects $k-1$ edges and removes them one by one under the constraint of network connectivity. The second offspring is generated by restarting the same process from the edges that have not been picked yet. The second crossover operator allows preserving the locations of sensors without transmitting the parents' communication links to offspring. Then, each sensor node in a parent must find its matching sensor node in another parent in order to form the edges of the offspring. Simulation results indicate that the first crossover operator produces better solutions than the second one. According to the authors, this could be due to the ability of the first operator to conserve the topological and geometric characteristics of parents. In [100], the authors tackled the problem of relay nodes placement into an existing static WSN. They assumed that the target area is a 2D rectangle without obstacles, and the sensor nodes are battery-powered devices. The solution is modeled by a vector containing the Cartesian coordinates of the relay nodes, and the fitness function comprises three objectives: energy consumption, coverage area, and network lifetimes. The classical versions of NSGA-II and SPEA2 evolutionary algorithms and MO-VNS trajectory algorithm were implemented and compared in terms of hypervolume and set coverage. Simulation tests showed that the NSGA-II is outperformed by the mentioned trajectory algorithm. Authors in [101] tackled the problem of deploying wireless sensor networks in smart buildings. They employed SPEA-II, NSGA-II, and NSGA-III to simultaneously optimize deployment cost, coverage, over-coverage mitigation, and network connectivity. The authors adopted basic binary sensing and communication models for sensor nodes

and did not account for obstacles within the buildings. Moreover, the three metaheuristics were evaluated on a simple small-sized indoor environment of 80 m^2 modeled by a grid without obstacles. In [102], the authors employed NSGA-II, SPEA-II, and other evolutionary algorithms to optimize WSN deployment with the objective of minimizing uncovered targets and maximizing available sensors. However, their strategy lacked precise modeling of sensor networks and the target area, and it did not consider the connectivity constraint.

Table 3.2: Main WSN deployment approaches based on NSGA-II

Work	Year	Objective	Constraints	Dimension	Obstacles
[56]	2020	Coverage Cost	-	3D	Homogeneous
[64]	2017	Coverage Cost Connectivity Over coverage	-	2D	Heterogeneous
[87]	2020	Coverage Cost	Connectivity	2D	Heterogeneous
[88]	2016	Coverage Network lifetime	Connectivity	2D	-

Cuckoo search algorithm (CS): CS is an evolutionary algorithm that has been applied in the literature to solve the WSN deployment problem [91, 94, 95]. Its pseudo-code is depicted in the Algorithm 2.

Algorithm 2: Pseudo code of the cuckoo search algorithm

```
Initialize population;
Compute the fitness of each cuckoo;
while Termination condition is not satisfied do
    Select a cuckoo i randomly by Lévy flights;
    Select a nest j randomly from the population;
    if fitness of cuckoo i > fitness of cuckoo j then
        Replace cuckoo i by cuckoo j;
    end
    Replace a fraction (P_a) of the worst solutions with new Keep the best solutions;
    Rank solutions and identify the current best;
end
```

In [91], the authors proposed a two-stage mobile sensor deployment approach based on the CS algorithm. This approach aims to ensure maximum coverage with a minimum

number of mobile sensor nodes and average mobile distance. Its first stage tries to maximize coverage by positioning the sensors in a target area digitized into a 2D grid without obstacles. For that, the authors used the CS algorithm with a Levy flight search mechanism to randomly select nests. Then the deployment scheme is optimized in the second stage by reducing the number of sensors and the moving distance. Another solution based on the CS algorithm was suggested in [94], the authors considered the coverage maximization of heterogeneous sensors network. Each cuckoo's egg is depicted as an array of the sensors' coordinates in a 2D area grouped by sensor types. The authors also improved the CS algorithm by adjusting the levy flight parameters; this allows creating new solutions with much smaller step lengths so that the new individuals will not be pulled away from the best solutions. According to the experimental results, the improved CS algorithm provides a good solution in a short time compared to other metaheuristics. In [95], authors developed an improved CS algorithm to tackle the k-target coverage problem of randomly deployed sensor nodes with adjustable sensing radius. The problem was formulated as a non-linear integer programming problem to optimize both coverage and network lifetime. For that, the authors assumed a corresponding energy consumption for each sensing range and divided the sensors into a set of non-disjoint covers activated alternatively to expand the network lifespan.

3.2.2 Swarm intelligence optimization algorithms

Swarm intelligence (SI) is another sub-field of artificial intelligence that has been widely applied to solve nonlinear problems related to several real-world domains [103]. SI is based on the collective behavior of agents. Each agent represents a solution to the problem and it adapts its behavior autonomously. Moreover, a SI system has two essential features, self-organization and labor division[104]. Self-organization characterizes a swarm's ability to grow over iterations via the interaction of its components without the need for external intervention, whereas labor division refers to the simultaneous execution of several tasks by the swarm's agents. There are common phases between EA and SI systems namely, population initialization, defining stop condition, and evaluating fitness function [105]. Yet, each SI algorithm has its own strategy for updating the movement of its agents. Fig. 3.1 illustrates the general framework of a swarm intelligence system.

Particle swarm optimization algorithm (PSO) PSO is a nature-inspired algorithm based on swarm intelligence [6]. The pseudo-code of PSO is depicted in Algorithm 3. Several works have applied PSO algorithm and its variants to find the optimal placements of WSNs [106, 107, 108, 109, 110, 69, 111, 112, 113, 114, 57, 115]. Author in [106], considered

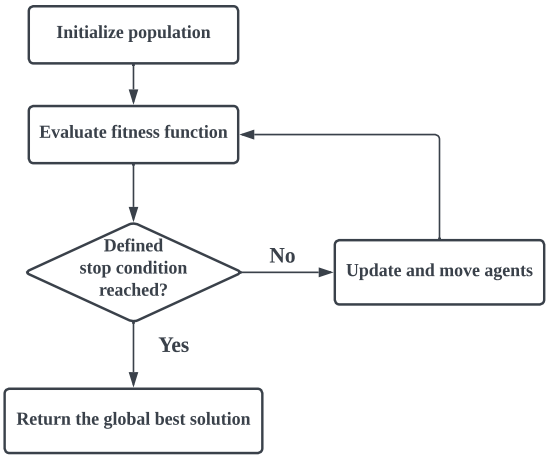


Figure 3.1: Swarm intelligence framework [105]

the WSN deployment problem in 3D terrain. To solve this problem, the author proposed a combination solution of the distributed particle swarm optimization and the 3D virtual force algorithm to maximize the coverage. The 3D virtual force algorithm helps avoid obstacles in the zone of interest and maintain network connectivity. Furthermore, the author has developed a heuristic to manage the communication limits of the sensor nodes by clustering the network so that each sensor node can communicate with the base station.

Algorithm 3: Pseudo code of PSO algorithm.

```
Initialize particles;
Compute fitness for each particle;
while termination condition is not satisfied do
    for each particle Particle_i in the swarm do
        Compute Fitness(Particle_i);
        Update Personal Best Fitness(Particle_i);
        Update Personal Best Position(Particle_i);
    end
    Update Global Best Solution();
    for each particle Particle_i in the swarm do
        UpdateVelocity(Particle_i);
        UpdatePosition(Particle_i);
        Update Personal Best Position(Particle_i);
    end
end
```

Authors in [107] proposed a PSO-based solution for deploying a WSN used by environmental and health applications. They intended to find the optimal locations of sensors

to optimize network coverage and lifetime while considering the connectivity constraint. The Minimum Spanning Tree (MST) routing protocol was applied to reduce the network's energy consumption, thereby extending its lifespan. *Qi et al.* in [108] studied the WSN redeployment problem to increase the network coverage and reduce the moving distance of mobile sensors. The proposed approach is based on the PSO algorithm and adapts a new nonlinear decreasing inertia weight as an improvement to avoid falling in local optima. Next, the deployment scheme is adjusted using the virtual force algorithm. *Li et al.* in [109] adapted the discrete binary particle swarm optimization for WSN deployment to optimize three objectives: network coverage, dormancy rate for boosting network lifetime, and coverage uniformity. The authors adjusted the particle velocity expression introducing a dynamic regulation of inertia weight, cognitive, and social factors. Also, they appended an escape operator that introduced a random position in the search space to avoid the local optima. Simulation experiments show that the proposed algorithm outperforms other solutions reported in the literature regarding the number of active nodes, coverage uniformity, and network energy consumption when the area coverage rate is more than 90%. *Yarinezhad et al.* [110], conceived a solution for WSNs deployment to solve the target coverage problem while considering the network lifetime. They used two versions of PSO: the cooperative PSO, which fits mainly the large-scale problems, and the cooperative PSO using fuzzy logic to adjust the acceleration factors dynamically. The simulation results demonstrate that the two suggested algorithms outperform GA, PSO, and the artificial bee colony technique in terms of network lifetime. Authors in [111] proposed an energy-saving solution for maximizing coverage in WSNs deployment. The solution is based on the PSO algorithm combined with an intelligent heuristic called Quasi physical force to decrease the overlapped coverage. Furthermore, the authors applied a dynamic balancing strategy to reduce energy consumption by shortening the sensing radius of some sensors in the network. *Ni et al.* in [112] tackled the problem of dynamic deployment, which aims at adjusting mobile sensors placements while considering both coverage maximization and moving distance minimization. They presented a heterogeneous multi-swarm PSO algorithm where the measurement of the traveling distance of a particle is computed using a discrete PSO, and the population is divided into three subswarms. Each swarm has different evolutionary strategies: PSO with inertia weight, PSO with constriction factor, and dynamic probabilistic PSO. This allows for boosting the population diversity and balances the exploration and exploitation of the algorithm. Furthermore, the authors compared the performance of their solution to the performance of two other PSO algorithms(classical PSO and co-evolutionary PSO), and the simulation results demonstrate that the multi-swarm PSO provided superior solutions with a reduced

moving distance and a maximum coverage rate than the two PSO methods. A variant of PSO called Social Class Multi-objective Particle Swarm Optimization (SC-MOPSO) was applied in [115] to deal with the problem of WSN deployment. It aims to minimize both the uncovered area and the deployment cost. The particles have variable lengths depending on the number of deployed sensors. SC-MOPSO splits the population into several classes where each class contains particles of the same length. The interaction between classes is done by moving the particles from the less-performing class to the higher one. According to the results, SC-MOPSO performs significantly better than other benchmarks in terms of dominating solutions. Table 3.3 summarises the main deployment approaches based on PSO.

Table 3.3: Main WSN deployment approaches based on Particle Swarm Optimization

Work	Year	Objective	Constraints	Dimension	Obstacles
[109]	2021	Coverage Sleep rate Coverage uniformity	-	2D	-
[106]	2020	Coverage	Connectivity	3D	Heterogeneous
[107]	2020	Coverage Lifetime	Connectivity	2D	-
[112]	2017	Coverage Reduce moving distance of mobile nodes	-	2D	-

Grey wolf Optimizer (GWO) GWO is another SI based algorithm that have been used to tackle the problem of WSN deployment [61, 116, 117, 118]. Authors in [61] developed an enhanced version of the GWO algorithm to deploy WSN in a 3D environment with the objective of coverage maximization under the connectivity constraint. For the first enhancement, the authors used the Tent map that generates chaotic research sequences, increasing population diversity and promoting algorithm exploration to escape the local optima. Another enhancement was the suggestion of a new position update strategy that splits the population equitably into an inner layer group to perform the inner layer encircle and an outer layer group to perform the outer layer encircle. The inner layer encircle focuses on the exploitation aspect of the algorithm and hence, impact the convergence speed. The outer layer encircle focuses on the exploitation aspect. The work in [117] focused on the coverage rate, the sensor nodes' distribution uniformity, and the average moving distance. They applied a Lévy-embedded Grey Wolf Optimization (LGWO) algorithm, which combines the GWO algorithm with the Lévy flight to enhance the searching mechanism and avoid the local optima. Additionally, the virtual force algorithm was introduced in sensor nodes' positions updating to maintain a connected network. Authors in [116] pro-

posed another approach based on GWO to tackle the problem of WSN deployment with three objectives: coverage rate, connectivity, and network energy. The proposed approach called a behavior-based grey wolf optimizer (BGWO), simulates two wolf groups' natural behaviors, namely the lost wolf strategy and the mating strategy. The lost wolf strategy allows the wolf pack to get rid of wolves with low fitness, and it is used to prevent the algorithm from falling in the local optima. In contrast, the mating strategy applies genetic operators to produce new individuals. Also, all the male wolves compete to mate instead of prioritizing only α wolves. A GWO variant called GWO-EH was presented in [119], with the objective of optimizing the WSN coverage. GWO-EH reinforces the exploration and exploitation processes through the improvement of the position-updating equation of the leading wolves and the repositioning of the worst three wolves around the leading wolves respectively. Furthermore, the hunting mechanism was also adjusted to involve the α wolves, β wolves, and γ wolves in the research process according to their ranks in the leadership hierarchy.

3.3 Hybrid metaheuristics

Despite their good performance in looking for near-optimum solutions in polynomial time, stand-alone metaheuristics still have limitations and drawbacks, such as premature convergence and low accuracy of solutions. This has motivated researchers to turn toward other optimization strategies based on the hybridization of algorithms [120]. Hybridization seeks to integrate two or more algorithms with complementary features to capitalize on and reap the benefits of their advantages [121]. The hybrid metaheuristics can be categorized into collaborative hybrids and integrative hybrids [122]. In collaborative hybrids, the combined algorithms work in multi-stage, sequentially, or in parallel as depicted in Fig. 3.2. For integrative hybrids, a subordinated algorithm is embedded in a master metaheuristic with a contributing rate between 10% to 20% [122]. Several hybrid-

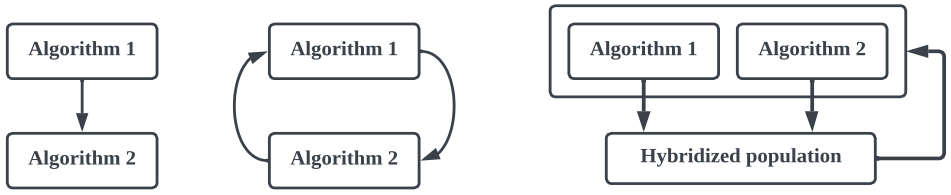


Figure 3.2: Collaborative framework of hybrid algorithm, depicting multi-stage, sequential, and parallel structures [122]

metaheuristics based approaches were designed to deal with the WSN deployment problem [123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134]. In the solution presented in [123], the authors conceived two hybrid algorithms, namely Hybrid-MOEA/D-I and Hybrid-MOEA/D-II, to simultaneously optimize three conflicting objectives: coverage rate, energy consumption, and equilibrium of energy consumption while positioning the sensor nodes in the target area. The last objective intends to limit the amount of energy consumed by a subset of sensor nodes in the WNS. For Hybrid-MOEA/D-I, the authors combined the MOEA/D framework with the three reproduction operators of GA and the differential evolutionary algorithm (DE): selection, crossover, and mutation. These operators are randomly selected for each sub-problem to increase the population diversity. The Hybrid-MOEA/D-II was developed by combining the Hybrid-MOEA/D-I with the discrete binary particle swarm optimization (DBPSO) to schedule sensor nodes and hence, boost the network lifetime. Another hybrid metaheuristic-based approach is proposed by *Mnasri et al.* in [124] to investigate the problem of finding the optimal 3D locations for additional nodes to an already deployed WSN. The approach combines the NSGA-III with the ACO to redress the low selection pressure problem of NSGA-III while maximizing coverage and keeping the ACO from falling into the local optima. ACO constructs the initial population of NSGA-III to produce only feasible solutions, and then for each iteration, the approach applies the classical steps of NSGA-III to create new solutions. These solutions are used in the next step to update the value of pheromones as a means to guide the search for fitter future solutions. A further hybrid method for WSN deployment is detailed in [125], it combines GA with binary ACO, which uses the binary coding of individuals. It also optimizes a mono objective fitness function that assesses the covered area and the number of working nodes. The initial population is randomly generated and enhanced using a repeated execution of genetic reproduction operators. Moreover, the new solutions are utilized to update information pheromones of the WSN. In the main loop, the algorithm executes the ants' traverse, updates the pheromone, and carries out the genetic crossover and mutation operators on the new solutions until the stopping criterion is met. In [135], the BA and the Grasshopper Optimization Algorithm (GOA) were hybridized to resolve the dynamic deployment problem of WSNs. The BA is known for its random behavior in both exploitation and exploitation phases. This reduces the algorithm's precision and convergence rate. To remedy these shortcomings, the authors applied the GOA algorithm in the exploitation phase to accurately exploit the neighborhood. GOA is a recent nature-inspired algorithm developed by Saremi et al. [136]. This algorithm mimics the behavior of grasshopper insect movement in searching for an optimal solution by aggregating the social interaction behavior, the gravity force factor, and

the wind advection factor in the same mathematical formula that computes the grasshopper position. Consequently, the hybrid BA changes each forager location depending on its current position, the position of the best solution in the neighborhood, and the position of all other foragers in the related neighborhood. This guides the BA search process to a more accurate solution within a reasonable convergence time. *Chen et al*. [130] conceived a hybrid framework based on an evolutionary algorithm called memetic algorithm and a heuristic recursive algorithm, designed to ensure a permanent full coverage with an extending network lifespan. Each potential solution of the memetic algorithm contains disjoint sets of sensor nodes that are sequentially activated using a scheduling mechanism. The heuristic recursive algorithm is developed to cope with the coverage hole problem caused by node failure or energy exhaustion through the activation of other nodes in other sets. The authors performed real-world tests to evaluate their approach and compared it with other existing solutions through computer simulations. The results revealed that the hybrid framework outperformed other algorithms in terms of network lifetime and this is for variant experimental conditions. Another hybrid solution was proposed in [131]. It combined the GA and the Binary PSO to compute the optimal deployment scheme with maximized coverage and connectivity and minimized cost. The hybrid algorithm began by creating the initial population and evaluating the fitness of individuals, then for each iteration, the population is spitted into two groups, the first group represents the input of GA and it encompasses the individuals with the highest fitness, and the worst solutions are sent to the binary PSO. With this solution scheme, the authors aimed at creating new solutions with evolved performances using the GA operators, and exploring other directions in the search space using the Binary PSO. The last step consists of merging the two outputs of GA and Binary PSO into one population for the next iteration. In the work presented in [132], the authors developed a hybrid search for the optimal WSN deployment based on the PSO and Hooke–Jeeves search method. The PSO is used to perform the global search in the search space. If the global solution is not improved after a preset number of iterations, the hybrid solution applies the Hooke-Jeeves method to carry out a local search in the neighborhood of the global best solution to improve its coverage and ensure faster convergence. *El Khamlichi et al*. [133], designed a hybrid approach based on gradient method and Simulated Annealing algorithm to deal with sensor nodes placement problem. The simulated Annealing algorithm is a local search metaheuristic that adopts the hill climbing moves to escape the local optima [137]. The hybrid approach has the objective of deploying the necessary number of sensor nodes to achieve at least 1-coverage and 1-connectivity, and it consists mainly of three major steps. The first step is to position the sensor nodes in the target area using a triangular grid deployment technique with a

preset distance between every two sensors, then in the second step, the gradient method is applied to reposition sensors placed on the area boundary to improve the network coverage. Finally, the connectivity constraint is ensured by adding new sensor nodes to fill the gap between connected and isolated sensors.

Table 3.4: Main WSN deployment approaches based on hybrid metaheuristics

Work	Year	Objective	Constraints	Methods	Dim	Obstacles
[135]	2021	Coverage	-	BA and GOA	2D	-
[124]	2017	Coverage	-	NSGA-II and ACO	3D	-
[133]	2017	Cost	Coverage Connectivity	gradient method Simulated Annealing	2D	-
[125]	2016	Coverage nodes utilization rate	-	GA and ACO	2D	-

3.4 Machine Learning

Machine Learning (ML) is a fundamental branch of AI that represents the intersection of computer science and statistics [138]. It enables computer systems to automatically learn from a huge amount of data and make predictions without being explicitly programmed for the task. The ML algorithms are classified into four categories based on the classification of the training data, these families are Supervised learning based on labeled data, Unsupervised learning based on unlabeled data, Semi-supervised learning based on a mixture of classified and unclassified data, and Reinforcement learning which does not require data. Several initiatives based on ML were proposed to deal with functional and nonfunctional aspects related to WSNs such as data aggregating, routing, localization, security, resource management, and sensor placement [139, 140, 141, 142, 143]. Authors in [144] developed an environmental sensor deployment algorithm based on a multi-response Taguchi-guided k-means clustering embedded GA. The deployment algorithm considers the coverage, connectivity, network lifetime, fault tolerance, and HVAC airflow optimization objectives, and it is conducted in three main stages. The initial stage of the deployment strategy is to determine the sensor locations that will provide an optimized network lifespan with a low installation cost. In the second stage, the network connectivity and the deployment cost of the relay nodes are addressed. Finally, the third stage considers the development of the entire system at the physical, network, and application layers with the aim of minimizing the total number the deployed sensor and relay nodes, while preserving the network performance. The multi-response Taguchi method has been applied to identify the best values for the crossover rate, the mutation rate, the

population size, and the number of clusters in k-means clustering. The k-means clustering is a machine learning method that aims at partitioning a set of observations into K clusters, with each observation belonging to the cluster with the closest centroid. It has been used in the deployment scheme to select the best cluster for the initial population with the best set of chromosomes in order to improve the convergence and computational time of the solution. The authors in [145] suggested a hybrid distributed approach for coverage hole healing based on game theory and Q-learning. Each mobile sensor node is depicted as a player that can compute its new position autonomously and in a decentralized manner. In their approach, the authors formulated the problem as a potential multiplayer game in which each player has to choose a combined action to improve simultaneously both coverage by reducing the overlapped zones and power consumption by adjusting the sensing radius and minimizing the motion energy. Further, the process of computing the payoff function is based on a multi-agent Q-learning algorithm since it involves the player's profile alongside the actions of its neighbors. The distributed payoff-based Q-learning algorithm is divided into two main phases. The first phase consists of selecting actions and updating states according to the exploration and exploitation processes and the second phase represents the learning phase which focuses on the other players' actions in order to react appropriately while repairing newly formed coverage holes. The simulation result shows that the distributed approach reached a better trade-off between the coverage and the energy consumption compared to other solutions proposed in the literature. Another work in [146] proposes a coverage hole detection and recovery approach based on a multi-intelligent agent-enabled reinforcement learning algorithm. In the first stage of the solution, the authors used the Sierpinski cluster-tree topology construction method to partition the sensor network into a set of unequal clusters. Then, they applied the multi-objective black widow optimization algorithm for the cluster head selection and a new Tsallis entropy-enabled Bayesian probability (TE2BP) algorithm for the dynamic scheduling of the sensor nodes. The goals of these previous phases are to ensure an energy-efficient transmission, minimize data loss, enhance energy consumption, and reduce the probability of coverage holes occurring. In the second stage of the solution, the virtual sector-based hole detection protocol is applied to detect the existing coverage holes in each cluster, then each coverage hole is healed using the multi-agent SARSA (State-Action-Reward-State-Action) algorithm. This algorithm takes as an input the coverage hole location and the list of mobile nodes around it and determines the optimal mobile node to heal the hole as an output. The factors that allow to SARSA algorithm to learn the environment are distance, node lifetime, and coverage level. The simulation tests show that the proposed algorithm outperforms existing solutions in the literature

in terms of coverage rate, the number of dead nodes, average energy consumption, and throughput.

3.5 Limitations

Existing deployment approaches in literature suffer from different limitations including:

- Several deployment approaches do not specify whether the deployment is intended for indoor or outdoor environments. Indeed, numerous factors differentiate the deployment of WSN in outdoor environments from those in indoor environments. For instance, indoor deployments must account for obstacles that obstruct sensor detection and communication signals, while outdoor deployments face interference from weather conditions. Additionally, the types of applications differ between indoor and outdoor environments, leading to variations in the choice of objective functions for optimization. For example, energy consumption becomes a crucial consideration for outdoor deployments, particularly in hostile areas, whereas it holds less importance for indoor sensor systems.
- The limited deployment approaches proposed to address indoor deployment often rely on oversampling the target area modeling. Typically, the target area is represented as a two-dimensional plane, often without obstacles or with a restricted range of uniform obstacles. However, indoor environments and smart buildings contain diverse obstacles constructed from various materials, significantly impacting the deployment process and network performance. Therefore, achieving an accurate representation of the building for sensor deployment is a crucial process to ensure reliable WSN deployment that can be effectively applied in real-world scenarios.
- Many deployment approaches based on metaheuristics fail to justify their choices of the selected metaheuristics. Indeed, since these metaheuristics have often been tested in very simple target areas with oversimplified network models, their real performance in addressing complex problems cannot be adequately assessed. Furthermore, the scalability of proposed methods is often not evaluated, as most methods focus on small and medium target areas. Therefore, conducting a comparative study is crucial to assess the true performance of metaheuristics before selecting the most suitable one for addressing the problem at hand. Such a comparative study should encompass various sizes of target areas with heterogeneous obstacles.

– Metaheuristic algorithms serve as powerful tools that provide a generic framework for tackling intricate real-world problems. Nevertheless, their design primarily focuses on general-purpose optimization rather than being customized for particular problems. Therefore, to enhance proposed deployment approaches in the literature, it becomes essential to integrate problem-specific heuristics or local search algorithms with problem-dependent rules.

– In real-world scenarios, ensuring connectivity among sensors is essential to avoid losing the collected data. However, this critical constraint is often neglected or considered as a objective function to be maximized.

– The modeling of WSN is often oversimplified and fails to reflect real-world settings. According to our survey in [5], more than 68% of reviewed approaches use a simple binary sensing model, while 63% do not specify communication range or rely on a simple binary communication model.

3.6 Conclusion

In this chapter, various WSN deployment solutions employing AI-based techniques, such as evolutionary algorithms, swarm intelligence optimization algorithms, hybrid metaheuristics, and machine learning, are reviewed and analyzed. Many of these deployment approaches prioritize enhancing multiobjective metaheuristics through hybridization or adjusting search mechanisms to enhance exploration and exploitation. However, they often neglect crucial aspects of WSN deployment, specifically network and target area modeling. Network modeling requires accounting for sensor characteristics while calculating their sensing and communication zones, while target area modeling involves considering all the features that may influence the sensor network topology and the effects of obstacles on the network. Oversimplification of these aspects results in a significant disparity between simulated and real-world performance of the sensor network. In the subsequent chapter, our proposed model for WSN deployment problem in smart buildings is introduced. It outlines our approach to target area modeling based on digital models and highlights the problem formulation. Additionally, our proposed heuristics for addressing connectivity constraints in sensor networks are discussed.

Part II

WSN Deployment in Smart Buildings

CHAPTER 4

PROBLEM MODELING

4.1 Introduction

Solving the WSN deployment problem within a smart building involves finding the optimal position for each sensor to optimize network coverage along with other objectives. In this thesis, we focus on optimizing target coverage instead of area coverage for the following reasons: firstly, we assert that not all locations within the building have equal significance and must be covered uniformly. Target coverage allows decision-makers to express their preferences by introducing target points in zones they wish to cover, aligning the deployment with specific needs. Secondly, the coverage area is complex to implement and requires a substantial amount of computational resources. This complexity arises from computing the union of all the detection zones of deployed sensors, which involves measuring the distortion of their detection areas based on their signal propagation in the surrounding environment and this is in all directions. Computing irregular shapes can be challenging and wasteful use of computational resources if the target area is not intended to be covered uniformly. Additionally, we introduce a target area modeling approach based on Building Information Modeling (BIM) since it incorporates all the necessary data for the optimization process. Finally, we discuss our proposed method to handle the connectivity constraint and generate interconnected Wireless Sensor Network (WSN) topologies.

4.2 Target area modeling based on BIM

The primary issue with the existing solutions in the literature is the lack of a reliable and comprehensive data source to model the target area, namely indoor environments. Researchers typically consider either a grid as a deployment zone or a 2D plan with insufficient data. The Building Information Modeling (BIM) represents a complete data source for obtaining building information. BIM models are based on an IFC file, often difficult for humans to read, where each entry represents an object or a relationship between objects within the building. Therefore, this section proposes a system capable of processing the IFC file and modeling the building in a suitable format usable by the optimization algorithm.

4.2.1 Building Information Modeling (BIM)

Building Information Modeling (BIM) has become a necessity for effectively managing projects that involve stakeholders from various disciplines within the Architectural, Engineering, and Construction (AEC) industry [147]. According to the US National Building Information Model Standard [148], the BIM is defined as "*BIM is a process for creating and managing information on a construction project throughout its whole life cycle. As part of this process, a coordinated digital description of every aspect of the built asset is developed, using a set of appropriate technology. It is likely that this digital description includes a combination of information-rich 3D models and associated structured data such as product, execution, and handover information*". The progression of the building life cycle, which includes design, implementation, maintenance, and demolition, is facilitated by BIM as illustrated in Fig 4.1. BIM serves as a tool for precisely defining the flows and balances of materials and techniques utilized in each of these phases. Furthermore, in addition to providing a 3D digital representation of actual buildings, BIM also offers geometric and non-geometric attributes that precisely describe all building components [149]. These attributes cover various types of information, such as functional details (e.g., installation duration and costs), semantic information (e.g., aggregation, containment, or intersection details about building components), and topological information, providing insights into the spatial relationships among these components within the building [149]. BIM software programs utilize an open data model called Industry Foundation Classes (IFC) to store all BIM data and facilitate interoperability among them [150].

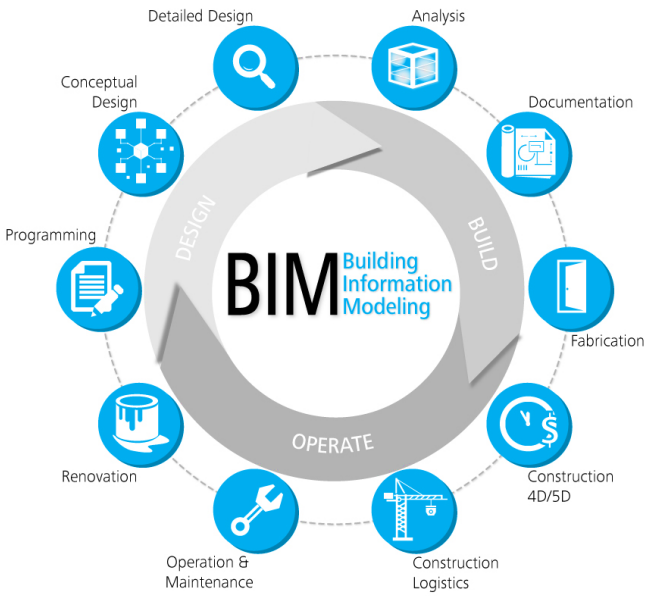


Figure 4.1: The utilization of BIM within the life cycle of infrastructure projects [151]

4.2.2 Industry Foundation Classes (IFC)

IFC is an open international standard certified by ISO (ISO 16739-1:2018) [152]. IFC serves as the predominant format for exchanging building information across AEC (Architecture/Engineering/Construction) software platforms. Its purpose is to seamlessly represent building data and facilitate data exchange between BIM modeling software applications. Within the IFC standard, building data is structured through classes, each inheriting from a common class known as IfcRoot. These classes can be categorized into three abstract classes, which are [153]:

- IfcObject: it encompasses all entities associated with construction activities, including physical objects (IfcProduct), actions (IfcProcess), or actors (IfcActor).
- IfcPropertyDefinition: it defines the various properties that can be assigned to an object within the building such as dimension and visual characteristics like color.
- IfcRelationship: it facilitates the connection between objects and other classes, whether they are properties or other objects.

All building data are represented using the IFC standard and stored in an IFC file, which is a plain text document. Each entry in the IFC file begins with the character sequence '#'.

These entries correspond to various building elements such as doors, floors, windows, etc., along with their associated attributes. Each entity is assigned a unique identifier and begins with the IFC class of the represented element, followed by its attributes (e.g., coordinates, dimensions, construction material), and relationships. Attributes and relationships may reference other entries within the same IFC file. Fig. 4.2 illustrates an excerpt from an IFC file.

```
#230= IFCSURFACESTYLE('Earth',.BOTH.,(#229));
#232= IFCSTYLEDITEM(#226,(#230),$);
#235= IFCSHAPEREPRESENTATION(#120,'Body','SurfaceModel',(#226));
#242= IFCSHAPEREPRESENTATION(#122,'FootPrint','Curve2D',(#222));
#245= IFCPRODUCTDEFINITIONSHAPE($,$,(#235,#242));
#251= IFCAXIS2PLACEMENT3D(#6,$,$);
#252= IFCLOCALPLACEMENT($,#251);
#253= IFCSITE('2xnuFRUG19DRdNdaDzqDaq',#42,'Surface:365763',$,$,#252,#245,$,.ELEMENT.,(42,21,31,181945),(-71,-3,-24,-263305),0.,$,$);
#258= IFCPROPERTYSINGLEVALUE('Reference',$,IFCIDENTIFIER('Surface'),$);
#266= IFCPROPERTYSET('3jAAUK5w53qf8mM8vGlimW',#42,'Pset_SiteCommon',$,(#258));
#274= IFCRELDEFINESBYPROPERTIES('3g6U6g2jz9N9H3QPk8oAf5',#42,$,$,(#253),#266);
#281= IFCAXIS2PLACEMENT3D(#6,$,$);
#282= IFCLOCALPLACEMENT(#184,#281);
#283= IFCCARTESIANPOINT((-29.2490874024323,303.));
#285= IFCCARTESIANPOINT((-29.2490874024323,320.499999999998));
#287= IFCCARTESIANPOINT((-236.12636889635,320.499999999999));
#289= IFCCARTESIANPOINT((-236.126368896352,-472.));
#291= IFCCARTESIANPOINT((-29.2490874024338,-472.));
#293= IFCCARTESIANPOINT((50.7509125975662,-472.));
#295= IFCCARTESIANPOINT((203.873631103648,-472.));
#297= IFCCARTESIANPOINT((203.87363110365,320.499999999997));
#299= IFCCARTESIANPOINT((50.7509125975677,320.499999999998));
#301= IFCCARTESIANPOINT((50.7509125975677,303.));
#303= IFCPOLYLINE((#283,#285,#287,#289,#291,#293,#295,#297,#299,#301,#283));
#305= IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,$,#303);
```

Figure 4.2: Caption from an IFC file

4.2.3 The proposed approach for modeling the target area

2D indoor plans are traditional and straightforward methods for depicting building layouts. While they provide a basic overview of the built environment, these plans often lack detailed information and require manual intervention to comprehensively model the environment. Alternatively, BIM models and their associated IFC files provide an effective means of describing indoor environments. Nevertheless, interpreting IFC files can pose challenges for humans due to their technical complex structure. In this section, we propose an approach for automatically modeling target areas based on their BIM databases. Our proposed method takes the IFC file describing the building as input and generates deployment zones, where sensors could be placed, along with corresponding building obstacles. The overall architecture of our BIM-based method for modeling target areas is depicted in Fig. 4.3. Our method comprises two primary components: the Data Extraction Component and the Modeling Component. Subsequently, we will provide a brief description of the function of each component.

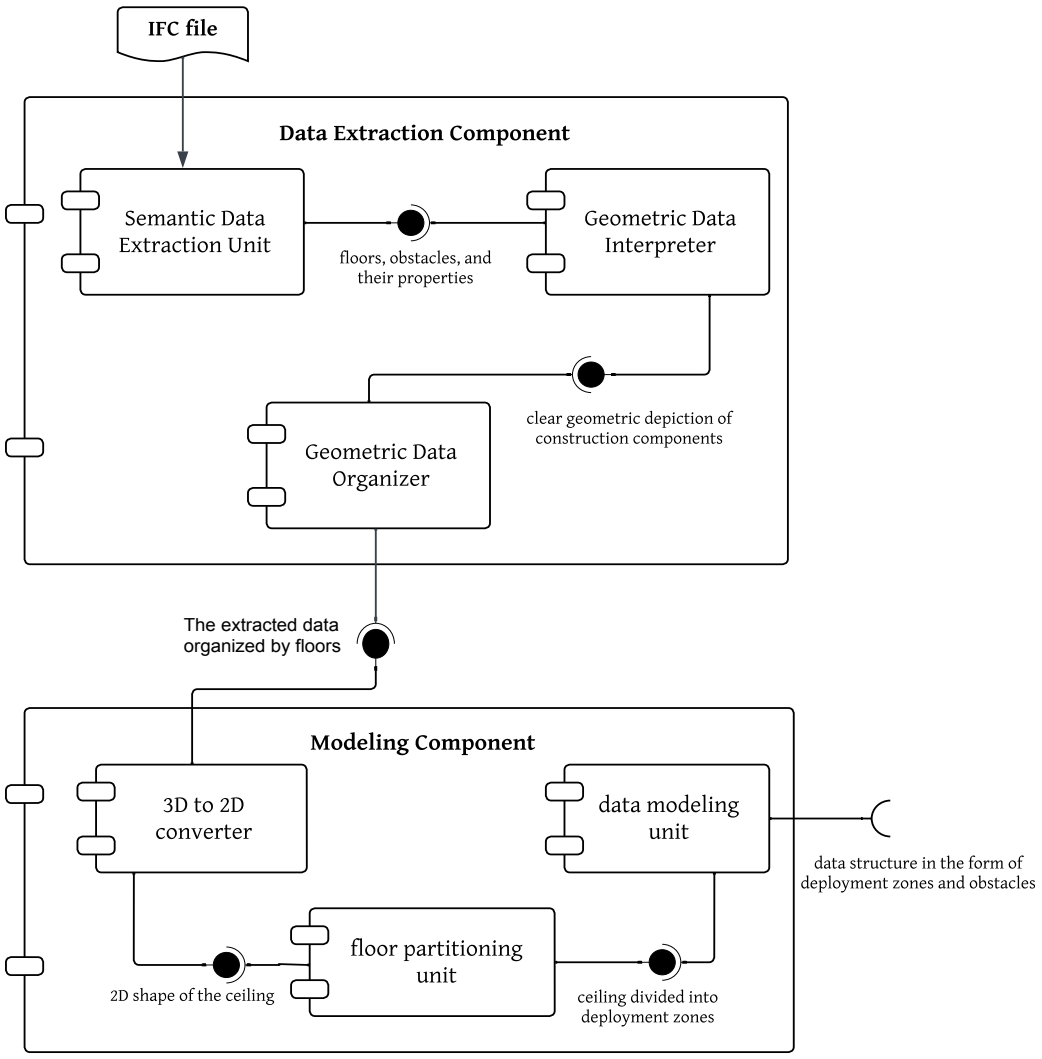


Figure 4.3: Global architecture of our developed BIM-based method for modeling target areas.

1. **Data Extraction Component:** It extracts from the IFC file all pertinent building data essential for the sensor deployment process. This information comprises the building's spatial organization (levels), the attributes inherent to these levels, and the geometric descriptions of building components. For each level (floor), the extraction component retrieves information regarding the ceilings and the presence of false ceilings and identifies obstacles, such as walls, curtain walls, windows, insulation, doors, and beams. Fig. 4.4 depicts the class diagram of this component. Its primary units include:

– Semantic data extraction unit: This unit is responsible for extracting IFC-standardized building obstacles (elements such as doors, walls, windows, etc) and their properties. It achieves this by conducting spatial queries on the designated dimension associated with

the respective floor. The distribution of obstacles according to their level is organized using a Bounding Volume Hierarchy (BVH)[1] tree structure. The geometric information related to obstacles is extracted in their raw form as described in the IFC standard. Therefore, the next unit is designed to handle these forms and convert them into manipulable forms.

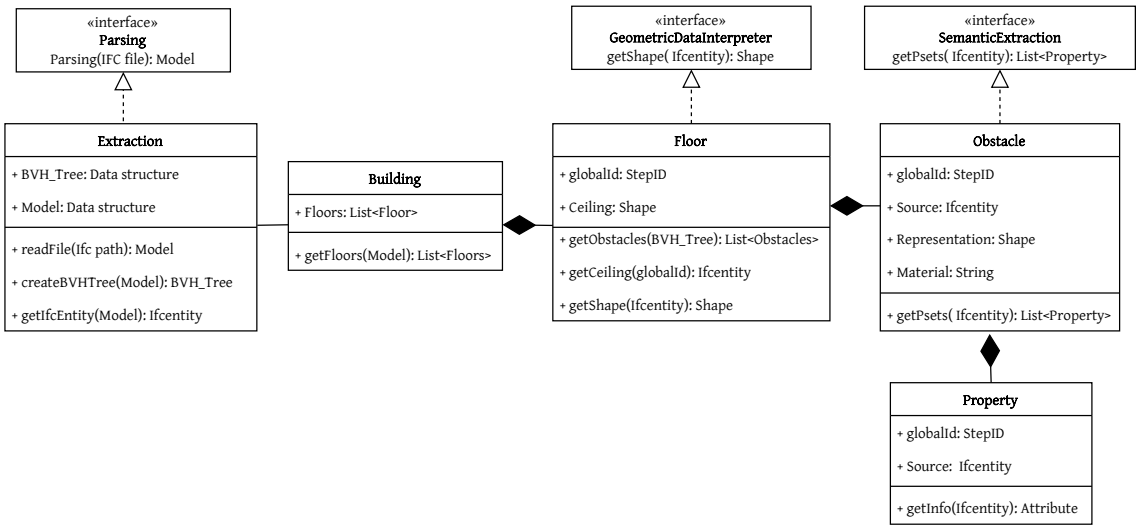


Figure 4.4: Class diagram of the Data Extraction Component.

- Geometric data interpreter: The IFC file stores geometric information in three forms, with Boundary Representation being the simplest form among them [154]. However, this form of representation is not readily usable by our system. Therefore, this unit transforms the geometric shape information of all obstacles into a more manipulable and usable representation called Faceted B-rep [155]. A Faceted B-rep is a simplified representation that depicts solid objects solely with planar facets and polygons. Consequently, any complex building obstacle with curved shapes will be approximated using flat facets.
- Geometric data organizer: It is used to arrange the extracted data into classes representing obstacles and their properties within a list that categorizes the building floors.

2. **Modeling Component:** The primary function of this component is to facilitate the optimization process by transforming the raw data generated by the Data Extraction component into a readily usable format. This approach prevents an infinite loop of computing continuous spatial coordinates of deployment points where sensors could be placed. To achieve this, we assume that sensors could be deployed only in false ceilings where available or in ceilings. Then, we cluster spatial points of this deployment area (false ceilings

[1]BVH is a tree structure that organizes geometric objects using nested bounding volumes. It is mainly used in computer graphics to accelerate spatial queries

or ceilings) into virtual distinct areas called deployment zones. All zones have almost the same size and are defined by a user-specified parameter x (x is an input that refers to the desired size of virtual deployment zones). The value of x can be determined based on the characteristics of the deployment area. If there are few obstacles that may obstruct the detection zone, x can be set close to the detection range of the sensors. However, in complex areas with numerous obstacles that could obstruct sensor detection, it is advisable to set x to a smaller value for more refined deployment. Here, the number of deployment zones is inversely proportional to the value of x; as x decreases, the number of zones increases, and vice versa. Fig.4.5 illustrates the class diagram of this component. Its key units consist of the following elements:

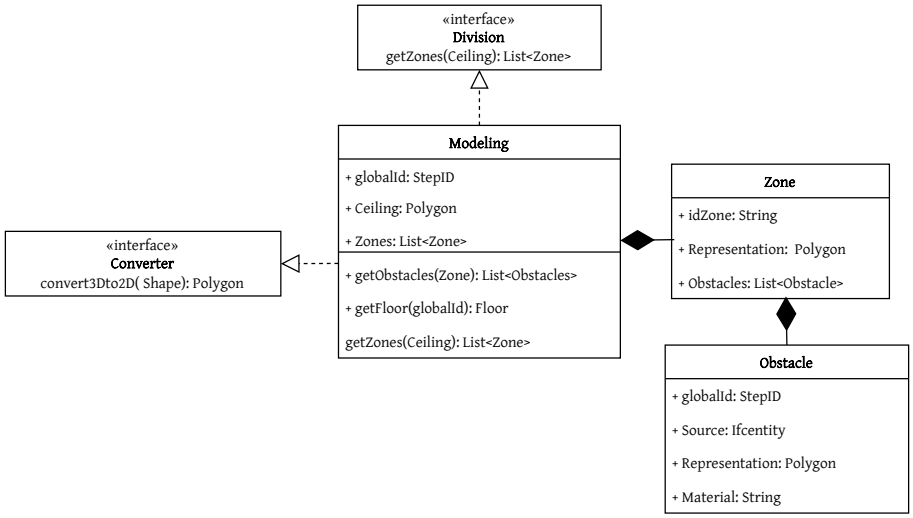


Figure 4.5: Class diagram of the modeling component.

- 3D to 2D converter: This module creates a top view of the floor where the WSN will be deployed and transforms the 3D ceiling into a 2D plan that can be partitioned into a set of deployment zones.
- Floor partitioning unit: This unit is responsible for partitioning the 2D plan of the false ceiling into virtual deployment zones. Initially, the floor (2D plan deployment area) is divided into two zones. If the size of these zones exceeds the predefined parameter x, they are further divided into two sub-zones. For each generated virtual zone, we recursively divide it either horizontally or vertically to ensure that the final deployment zones are of regular shapes (polygons). The division process stops when all deployment zones meet the constraint on their size. Fig. 4.6 illustrates an example of partitioning the first floor of the provided BIM model. The empty areas within the partitioning signify restricted zones where sensor deployment is not feasible, such as stairwells. Consequently, the centroids

of the resulting deployment zones indicate potential locations for sensor placement.

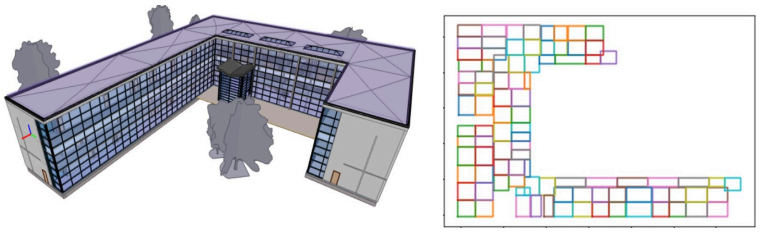


Figure 4.6: Example of a floor partitioning

– Data modeling unit: After partitioning the ceiling into deployment zones and obtaining a list of obstacles of the floor being modeled, this unit assigns each obstacle to its corresponding deployment zone.

4.3 Problem formulation

As mentioned earlier, the objective of this thesis is to propose metaheuristic-based approaches for deploying WSNs in smart buildings. As discussed in Chapter 1, sensors within smart buildings can be categorized into two types: contact sensors and non-contact sensors. Contact sensors lack a detection zone, and their measured data are directly correlated to their local positions. Therefore, these sensors require precise positioning, strategically calculated as previously discussed.

In this thesis, our focus is on deploying non-contact sensors, which have a detection zone. These sensors are primarily designed for occupancy detection and motion sensing, with a wide range of applications including energy efficiency, smart lightning, activity recognition, and fall detection used in smart elderly care applications.

To address this objective effectively, we model the WSN as a collection of fixed sensors $S = \{s_1, s_2, ..., s_n\}$, $n \in \mathbb{N}$. All sensors within this network share common characteristics including sensing range R_s, communication range R_c, and transmission power P_t. Additionally, the deployment area represents one floor of the building and it is modeled, as explained in Section 4.2, by a set of static obstacles $O = \{o_1, o_2, ..., o_p\}$, $p \in \mathbb{N}$ and a finite number of deployment zones $Z = \{z_1, z_2, ..., z_m\}$, $m \in \mathbb{N}$, depicting potential places where sensors can be deployed. Each obstacle is characterized by its shape, location, and construction material. The deployment area has also a finite number of target points $T = \{t_1, t_2, ..., t_q\}$, $q \in \mathbb{N}$ to be covered by the WSN.

The objective of the WSN deployment problem tackled in this thesis is to create interconnected network topologies within indoor structures that optimize simultaneously the network coverage and the deployment cost.

4.3.1 Solution encoding

After partitioning the deployment area into a set of deployment zones, these zones are then flattened row by row to assign a unique identifier to each zone. The first zone corresponds to the first zone in the deployment area, located at the beginning of the first row, and the last zone corresponds to the last zone in the deployment area, situated at the end of the last row. Consequently, we model the WSN topology using a binary vector $V = \{x_1, x_2, ..., x_m\}$ of a length equal to $|Z|$. Each decision variable x_i within the vector denotes the sensor presence ($x_i = 1$) or absence ($x_i = 0$) in the deployment zone $z_i \in Z$. In the case of continuous optimization algorithms, each decision variable x_i is discretized after each position update according to the binarization technique introduced in [156] and depicted by Eq. 4.1, unless another method for discretization is explicitly specified. This transforms the search space from a continuous domain to a binary one.

$$x_{i_{discretized}} = \begin{cases} 1, & sigmoid(x_i) \leq rand \\ 0, & otherwise \end{cases} \tag{4.1}$$

$$sigmoid(x) = \begin{cases} 1, & \frac{1}{1+e^{-10\times(x-0.5)}} \leq rand \\ 0, & otherwise \end{cases} \tag{4.2}$$

4.3.2 Objective functions

Given that our thesis aims to address the deployment problem of non-contact sensors in smart buildings, optimizing network coverage emerges as the primary objective. Furthermore, a significant portion of sensors in this category relies on costly technologies [24], emphasizing the critical importance of reducing deployment cost. Reducing energy consumption and maximizing sensor lifespan are very correlated objectives, both necessitating an increase in the number of deployed sensors. These objectives become particularly

crucial when deploying sensors in hostile environments where accessibility for sensor replacement in case of failure is limited. Thus, both objectives entail augmenting the number of deployed sensors to ensure prolonged network operation and prevent breakdowns caused by energy depletion or sensor failures.

In the context of deploying sensors in smart buildings, this issue isn't significant as replacing a node or its depleted battery is straightforward. Therefore, our thesis concentrates on two primary objectives for our deployment strategy: maximizing network coverage and reducing deployment costs. We've opted not to prioritize minimizing energy consumption and maximizing sensor lifespan since they hold lesser importance compared to reducing deployment costs.

The type of coverage addressed in this thesis is target coverage for the reasons outlined in Section 4.1. Hence, coverage optimization involves maximizing the number of covered targets within the deployment area. Similarly, deployment cost minimization refers to reducing the number of deployed sensor nodes.

Due to the complex nature of the WSN deployment problem, particularly in large smart building areas, we initially attempted to simplify it by merging the coverage and cost objectives into a single objective. This was achieved by aggregating these objectives into a single fitness function, as shown in Equation 4.3. Consequently, we propose mono-objective approaches based on mono-objective metaheuristics to solve it as it will be elaborated in Chapter 5. Besides simplifying the problem and minimizing computational costs, the primary advantage of initially adopting mono-objective approaches is to identify inherent conflicts and dependencies between the two objectives.

$$Min(\frac{1}{|Z|} \times \sum_{i=1}^{|Z|} x_i + \frac{1}{|T|} \times (|T| - \sum_{i=1}^{|T|} covered(S, t_i))) \tag{4.3}$$

Where:

$$covered(S,t) = \begin{cases} 1, & \exists s_i \in S, P_{det}(s_i, t) \geq threshold \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

Here, $P_{det}(s_i, t)$ represents the sensing model used to assess the detection range of sensors. We opted to use the Elfes sensing model [157], as defined by Eq. 4.5, based on the comparison between sensing models presented in Table 2.1.

$$P_{det}(s_i,t) = \begin{cases} 1, & d(s_i,t) \leq R_1 \\ e^{\lambda(d(s_i,t)-R_1)^\gamma}, & R_1 < d(s_i,t) < R_s \\ 0, & d(s_i,t) \geq R_s \end{cases} \tag{4.5}$$

R_s is the sensing range of the sensor, R_1 is the certainty detection zone, λ and γ depend on the physical characteristics of the sensor. The first component of Eq. 4.3 is dedicated to reducing the number of deployed sensors, while the second component focuses on minimizing the count of uncovered target points.

While mono-objective approaches can provide satisfactory solutions within a reasonable amount of time, they also confirm the inherent conflict between coverage and cost objectives. This conflict limits their ability to explore the entire search space. In complex real-world problems such as WSN deployment, decision-makers typically require a set of Pareto-optimal solutions rather than a single solution to select the best option according to their preferences. These Pareto sets comprise all non-dominated solutions, representing the solutions in the search space where optimizing coverage would result in increased cost. Hence, these sets offer decision-makers the best trade-offs between the conflicting objectives present in the search space. Therefore, in our second contribution, we propose a multi-objective approach based on multi-objective metaheuristics to efficiently solve the problem of WSN deployment in smart buildings and generate a set of Pareto-optimal solutions with a good trade-off between coverage and cost as explained in Chapter 6. The problem is formulated in this case as depicted in Equation 4.6.

$$Min\ [f_{cost}(V),\ f_{uncoveredTargets}(S,T)] \tag{4.6}$$

Where: f_{cost} is the deployment cost function defined in Eq. 4.7

$$f_{cost}(V) = \sum_{i=1}^{|Z|} x_i \tag{4.7}$$

and $f_{uncoveredTargets}$ is the coverage function defined in Eq. 4.8

$$f_{uncoveredTargets}(S,T) = |T| - \sum_{i=1}^{|T|} covered(S,t_i) \tag{4.8}$$

$covered(S, t_i)$ is defined by Eq. 4.4.

4.3.3 Network model

The WSN is modeled by an undirected graph $G = (V, E)$, where the vertex set V represents sensors positioned at deployment points within Z, and the edges set E signifies the communication links between these sensors. The estimation of communication probability is derived from the following equation:

$$P_{com}(s_i, s_j) = \begin{cases} 1, \text{ if } P_r(s_i, s_j) \geq sensitivity \wedge \quad d(s_i, s_j) \leq R_c \\ 0, \text{ otherwise} \end{cases} \tag{4.9}$$

Where $d(s_i, s_j)$ is the distance between sensor s_i and sensor s_j, *sensitivity* is the minimum signal strength a sensor can manage, and $P_r(s_i, s_j)$ represents the signal strength of the received communication signal between s_i and s_j, calculated as follows:

$$P_r(s_i, s_j) = P_t - PL(s_i, s_j) \tag{4.10}$$

Where P_t is the transmission power of the sensor node and $PL(s_i, s_j)$ is the path loss between s_i and s_j. This study used the Multi-Wall path loss model [158] defined by Eq.4.11:

$$PL(s_i, s_j) = PL(d_0) + 10 \times \eta \times log_{10}(d(s_i, s_j)) \tag{4.11}$$

$PL(d_0)$ signifies the path loss occurring at distance d_0, while η denotes the path loss exponent, which defines the propagation environment.

4.4 Connectivity constraint handling

During the generation of WSN topologies, the presence of obstacles inside the indoor environment may cause the network to fragment into separate segments of sensors. This fragmentation disrupts communication between these segments and leads to a loss of data collected by isolated sensor partitions. In Section 2.4, we discussed three methods for addressing the connectivity constraint during the optimization process: imposing penalties on unconnected topologies, adapting optimization method operators to exclusively generate connected topologies, and repairing the connectivity of unconnected topologies dur-

ing each iteration. In this thesis, we opt for the third approach to connectivity handling, as it overcomes the limitations of the preceding methods. Therefore, To address this connectivity issue, we have developed a connectivity repair heuristic in [159] to repair sensor topologies in case of multiple disjoint partitions. We propose two heuristics based on the Dijkstra algorithm and minimum Steiner tree respectively, to deploy the minimum number of additional nodes while preserving the initial topology.

The problem of WSN connectivity healing in the case of multiple disjoint sets has been widely addressed in literature [160]. Each approach has different assumptions regarding the type of network (stationary WSN, mobile WSN, or hybrid WSN) and the type of deployed nodes between different segments. Several techniques based on the relocation of mobile nodes have been proposed to eliminate the need for deploying new sensors for mobile and hybrid WSNs. On the other hand, methods based on the deployment of additional nodes have the objective of restoring connectivity at a minimal cost. In [161], authors suggested a connectivity healing protocol after multiple node failures. The protocol is based on the repositioning of mobile nodes to link segments. It involves partitioning the sensors into a set of disjoint connected segments. Each segment elites a leader to compute its position relative to the center of the deployment area. Redundant mobile nodes will be selected based upon their energy levels to populate the short path between the segment and the area center. In [162], *Lee et al.* have introduced DORMS, a distributed autonomous connectivity repair heuristic. DORMS intends to compute the least number and the position of relay nodes through three sequential phases. In the first step, a short path is built between each segment and the deployment area center using mobile relay nodes located within the segment, separated by a distance equal to the communication range. The next step aims at reducing the number of deployed relay nodes by computing the minimum Steiner tree (MST) between each two neighboring short paths and the center deployment area and then reassigning the extra nodes to their corresponding sets. In the final step, the best MSTs are retained and the relay nodes are relocated to fit the final scheme. The same authors developed in [163] a two-stage algorithm called ORC to identify the number and position of the required relay nodes to reestablish the WSN connectivity. In the first stage, the terminals (disjoint sets) in the border of the deployment area are identified using a convex hull. Then, a Steiner point (SP) is computed for every three adjacent terminals. Upon connecting all border terminals, the process is re-executed on the remaining terminals and designated SPs until all the terminals are connected. Next in the second stage, ORC populates the first-stage topology using relay nodes with respect to the communication range. This method has been enhanced in [164] to ensure fault tolerance through bi-connected topology. An alternative version of this method was presented

in [165] in which the Fermat point (FP) is substituted for the SP if the FP is located inside the triangle formed by the three terminals; otherwise, the Fermat point is substituted for the centroid of the triangle. *Liu et al.* [166] designed a distributed protocol to reestablish the connectivity after each single node failure. It divides the network into critical and non-critical nodes, each having its gradient values that determine which node will be the optimal backup to the damaged node. Authors in [167] proposed a heuristic based on Virtual Force and Compromise Strategy (CVFCS) to restore connectivity between multiple disjoint sets. The proposed heuristic runs in two stages. The first stage consists of adjusting the positions of sensors using Virtual Force to minimize the distance separating the disjoint sensor sets. The second step involves deploying a minimal number of relay nodes between the sets using minimum spanning tree and Steiner minimal tree algorithms.

Lalouani et al. in [168] argued that representing segments by single points as terminals is inefficient due to the lack of precision regarding segment shapes and sizes. Thus, they have proposed the Boundary-aware optimized Interconnection of Disjoint segments (BIND) method. This method represents segments as simple polygons delimited by the convex hulls of the border nodes. By using these border nodes as terminals, BIND seeks to construct the shortest total length tree in the Euclidean plane that links a subset of terminals so that all segments are connected. For this purpose, the Steiner minimum tree (SMT) is first generated for all boundary nodes. The edges of the SMT are then retained and used to form a minimum cost tree covering all segments.

Published approaches for restoring WSN connectivity primarily rely on pre-deployed mobile nodes to bridge the gap between sensor segments. Yet, these approaches cannot be applied to stationary WSNs. Moreover, new positions of mobile nodes are mostly computed using the minimum Steiner tree (MST) in order to reduce the number of involved nodes. The MST is calculated in the Euclidean plane (continuous space) with no restrictions on the new positions other than the boundary constraints. Nevertheless, in real-world situations, there could be various obstacles present in the deployment area that limit the Euclidean plane to a finite number of potential deployment sites. A relocation of mobile nodes deployed within a connected segment may result in a loss of connectivity within that segment as well. In addition, some scenarios require maintaining the overall WSN topology since the former positions of nodes were computed in accordance with other constraints such as coverage and network lifetime. Thus, the connectivity repair process should complete the deployment scheme with new nodes without significant changes to other segments. To cope with the aforementioned shortcoming, we developed two connectivity repair heuristics for stationary WSNs that preserve the previous WSN deployment scheme. In what follows, we will define the problem connectivity repair for

sensor networks and detail the proposed heuristics.

4.4.1 Problem formulation

We assume that the unconnected topologies are divided the network into disjoint sets of sensors $D = \{d_1, d_2, ..., d_m\}$, $m \in \mathbb{N}$. Each disjoint set d_i, $0 \leq i \leq m$ is defined as follows:
$d_i = \{s_k | k \in [0, ..., n] \wedge |d_i| < n \wedge \forall (s_j, s_l) \in (d_i \times d_i), P_{com}(s_j, s_l) = 1 \wedge \forall s_j \in d_i, \forall s_l \notin d_i,$
$P_{com}(s_j, s_l) = 0\}$

The problem of connectivity repair of WSN tackled here, consists of deploying the minimum number of additional sensor nodes in deployment zones so that the WSN will be connected. Therefore, the solution to this problem will select the minimum number of adequate deployment zones where the additional nodes will be placed to restore the network connectivity.

The problem could be defined as follows :

$$\text{Minimize } \sum_{i=1}^{q} z_i$$

Subject to $|D| = 1$.

4.4.2 Proposed connectivity repair heuristics

As mentioned before, our proposed heuristics aim to restore the connectivity in a WSN by adding a minimum number of additional nodes. Moreover, the two methods are based on the Dijkstra short path algorithm and the Steiner tree algorithm and are destined for obstacle-free areas and obstacles-filled areas respectively. The two approaches are split up into two stages in order to reduce their complexity. The first stage is the same in both heuristics and consists of connecting disjoint sets of deployed sensors having at least one shared zone in their neighboring deployment zones. To do so, the deployment zone z having the highest occurrence in the neighboring deployment zones of all disjoint sensor sets is selected at each iteration then, a new sensor is deployed at z and the set of disjoint segments is updated by merging new connected segments. The process is repeated until all zones have a number of occurrences equal to one, which indicates that there are no shared zones between the disjoint segments. Algorithm 4 depicts the first stage of connectivity repair heuristics.

Based on the deployment area characteristics, we select the appropriate method to repair the network connectivity as explained below.

Algorithm 4: Merge disjoint sets of sensors having a shared zone in neighboring deployment zones

Data: Z /*list of zones*/
Ind /*infeasible solution*/
Result: D: /*list of disjoint sets having no shared zone in their neighboring deployment zones*/
S: /*list of the neighboring deployment zones of sets in D */

```
S ← ∅;
D= Create Disjoint Sets Of Sensors from Ind;
for i ← 0 to |D| do
    Create set s_i containing all deployment positions reachable from d_i ∈ D;
    S ← S ⋃ s_i;
end
continu ← True;
while continu do
    P = Union of all s_i ∈ S;
    l = Deployment position with the highest number of occurrences in P;
    if l > 1 then
        Merge all sets d_i ∈ D having position l in their s_i;
        Deploy a sensor at position l and add it to the merged set;
        Update D, S ;
        /*remove the disjoint sets from D and replace them with one set that merges
          them all */
    else
        continu ← False;
end
Return D, S ;
```

Obstacle-free target area: For this method, it is assumed that the deployment area is free of obstacles or that obstacles have a limited impact on communication signals. Initially, the proposed method applies Algorithm 4 to connect the closest segments with shared neighboring zones. After the first stage, a minimum of two new nodes should be deployed to link a pair of the remaining disjoint segments.

To achieve full network connectivity with minimum additional nodes, a second stage process based on the Dijkstra algorithm is proposed. First, the central point (x_i, y_i) of each set of neighboring zones S of the remaining disjoint segments is computed using the formula below and considered as a terminal.

$$(x_i, y_i) = \min_{i \in [1,\dots|S|]} \sum_{j=1, j<>i}^{|S|} \frac{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{|S|}$$

Next, the terminals are organized in pairs and sorted according to the ascending order with respect to the distance between them. Then, the pairs of terminals are selected one at a time in the aforementioned order and the Dijkstra algorithm is applied to the communication graph of the deployment zones to compute the short path between them. The vertices of the short path represent the minimum deployment zones where additional sensors should be placed to establish the connectivity between the two disjoint segments corresponding to the selected pair of terminals. This process is repeated until the network is fully connected. Algorithm 7 depicts the steps of the proposed heuristic.

Target area with obstacles: The short path-based method is not effective when there are significant obstacles that completely block communication signals in the deployment area. The reason for this is that closest sets in terms of Euclidean distance could have blocking obstacles in their line of sight, thus, short paths will bypass these obstacles, resulting in an increased number of additional nodes. To cope with the problem, we proposed a second heuristic depicted in Algorithm 5 for obstacles-filled areas to connect terminals. The present method has the same steps as the short path-based method. Yet, the short paths between segments are computed using the Steiner tree algorithm depicted in Algorithm 6 of the communication graph between deployment zones induced by terminals.

Algorithm 5: The proposed Steiner tree based connectivity repair heuristic

Data: Z /*list of zones*/
Ind /*infeasible solution*/
Result: $Ind_{connected}$: /*feasible solution*/
$D, S \leftarrow$ Merge Disjoint Sets using Algorithm 1;
Update Ind according to D;
$graph$ = Create a graph representing connections between all the zones;
$terminals \leftarrow \emptyset$;
for $s \in S$ **do**
  Append the most centered point of s to $terminals$;
end
$tree \leftarrow$ Steiner tree approximation heuristic (see Appendix A) of $graph$ induced by $terminals$;
Delete all edges from $tree$ whose vertices are in the same set $s \in S$ while keeping $tree$ connected;
$Ind_{connected} \leftarrow$ Place new sensors in vertices of $tree$ and add them to Ind;
Return $theInd_{connected}$;

Algorithm 6: Steiner tree approximation heuristic

Data: G /*graph*/
$Terminals$ /*list of terminals*/
Result: ST: /* approximation of Steiner tree of G induced by $Terminals$*/
$Metric \leftarrow$ the metric closure of G /*$Metric$ is the complete graph induced by G in which the edge weight between each pair of nodes represents their short path distance*/;
$Sub \leftarrow$ the subgraph of $Metric$ whose vertices represent $Terminals$;
$ST \leftarrow$ the minimum spanning tree of Sub;
Return ST ;

Algorithm 7: The proposed Short path based connectivity repair heuristic

Data: Z /*list of zones*/ Ind /*infeasible solution*/
Result: $Ind_{connected}$: /*feasible solution*/
$D, S \leftarrow$ Merge Disjoint Sets using Algorithm 1;
Update Ind according to D /*deploy additional nodes*/ ;
$graph$ = Create a graph representing connections between all the zones;
$terminals \leftarrow \emptyset$;
for $s \in S$ **do**
    Append the most centered point of s to $terminals$;
end
$DistanceBetweenTerminals \leftarrow \emptyset$;
for *each* $t_i \in terminals$, $t_j \in terminals$, $i <> j$ **do**
    Append $(t_i, t_j, Distance(t_i, t_j))$ to $DistanceBetweenTerminals$;
end
Sort $DistanceBetweenTerminals$ in ascending order with respect to distance;
$i \leftarrow 0$;
$tree \leftarrow$ empty graph;
$insertedTerminals \leftarrow \emptyset$;
$t_1 \leftarrow DistanceBetweenTerminals$[i][0];
$t_2 \leftarrow DistanceBetweenTerminals$[i][1];
while *length(insertedTerminals)* $<$ *length(terminals)* **do**
    if t_1 *or* t_2 *not in insertedTerminals* **then**
        Compute the short path in $graph$ between t_1 and t_2 using Dijkstra algorithm;
        Add all the edges of the short path to $tree$;
        Update $insertedTerminals$;
    end
    $i \leftarrow i + 1$;
    Update t_1 and t_2 ;
end

Algorithm 8: The continuation of the proposed short path heuristic

$D \leftarrow$ Distinct connected sets of *tree*;
Paths $\leftarrow$ pairs of terminals in *DistanceBetweenTerminals* not inserted in *tree*, in ascending order with respect to distance;
while $|D| <> 1$ **do**
    for *each path in paths* **do**
        if *path links two disjoint sets in D* **then**
            Compute the short path in *graph* between $path[0]$ and $path[1]$ using Dijkstra algorithm;
            Add all the edges of the short path to *tree*;
            Update *insertedTerminals*;
            Update *D*;
        end
    end
end
Delete all edges from *tree* whose vertices are in the same set $s \in S$ while keeping *tree* connected;
$Ind_{connected} \leftarrow$ Place new sensors in vertices of *tree* and add them to *Ind*;
Return $Ind_{connected}$;

Figure 4.7 shows the deployment zones selected by both proposed heuristics for placing the additional sensors.

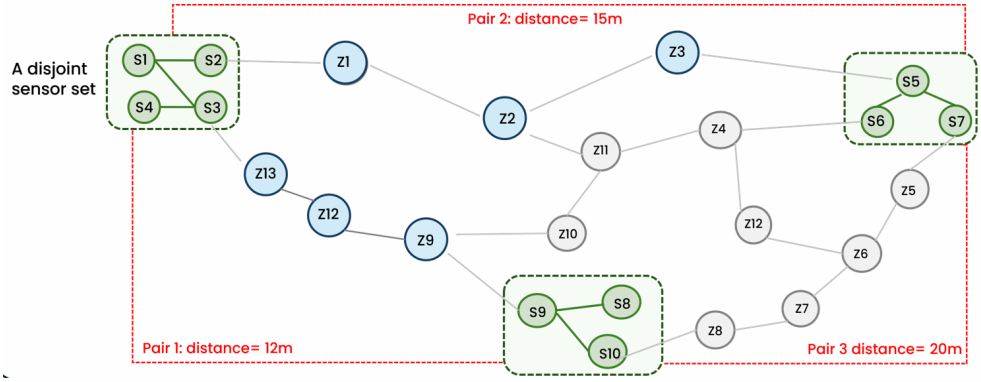


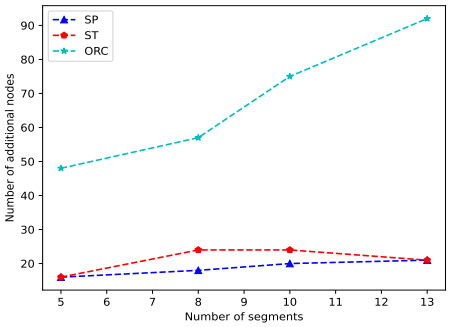
Figure 4.7: Selected deployment zones by the proposed heuristics.

4.4.3 Performance evaluation

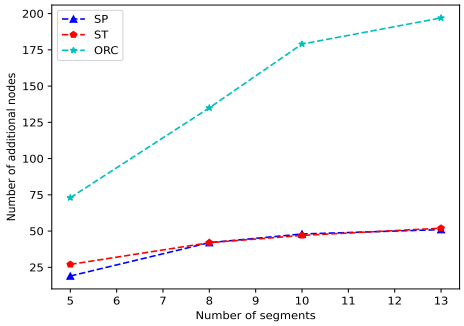
This section presents simulation results to evaluate the performance of our two proposed methods for WSN connectivity repair. The two heuristics have been tested against the

number of additional nodes and execution time and compared with the ORC approach (explained in Section 4.4). All the algorithms were implemented and tested in Python 3.7.8. For these tests, we have considered that all the sensors are omnidirectional with a communication range of 7m. Only areas that are free of obstacles have been considered in order to test all algorithms under the same conditions. For that, we examined 12 scenarios where the size of the deployment area is fixed to $80 \times 80m^2$, $100 \times 100m^2$, and $120 \times 120m^2$, and for each deployment area, we generated four random WSN topologies of 5, 8, 10, and 13 disjoint segments.

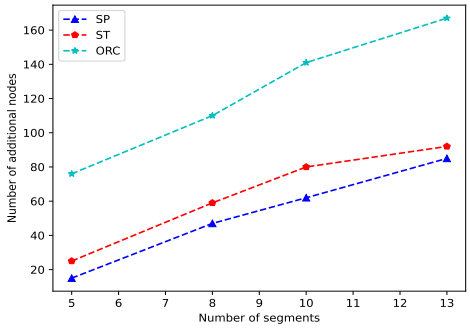
Number of additional sensor nodes: Fig 4.8 illustrates the number of additional nodes required to restore connectivity in each scenario. For the sake of clarity, we referred to the short path-based method as SP, the Steiner tree-based method as ST, and the ORC approach as ORC. As shown in Fig 4.8a, Fig 4.8b, and Fig 4.8c both SP and ST outperform ORC in all scenarios. We notice a significant difference in the number of additional nodes when a topology contains a greater number of disjoint sensor segments.



(a) Number of additional nodes vs number of segments ($80 \times 80m^2$)



(b) Number of additional nodes vs number of segments ($100 \times 100m^2$)



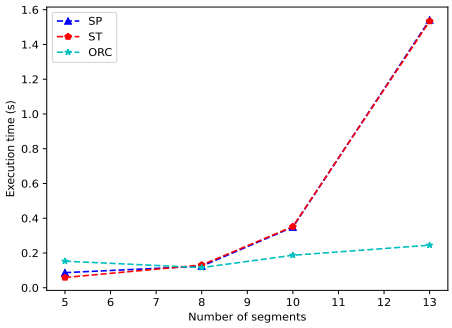
(c) Number of additional nodes vs number of segments ($120 \times 120m^2$)

Figure 4.8: Number of additional nodes as a function of segments number and deployment area size

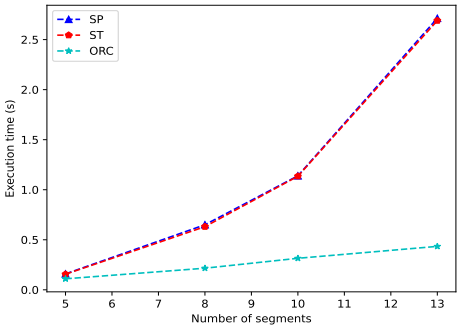
For example, the required number of additional sensors in the case of $100 \times 100m^2$ area

with 13 segments is 51 for SP and 52 for ST yet, ORC has a number of additional nodes of 197. Further, it is apparent that SP and ST have almost the same performance in all scenarios, for instance, with 13 segments in $80 \times 80m^2$, both methods required 21 additional nodes, while for $100 \times 100m^2$ area, they have almost the same number of additional nodes for all number of segments. This is because both algorithms are tested in free obstacle areas, and the short paths used to construct the Steiner tree by ST are most probably linking the closest sets in terms of Euclidean distance, resulting in a similar outcome to SP. Using ORC, disjoint sets are linked by iteratively connecting them in a circular manner until the center point of the deployment area is reached. This results in a significant increase in the number of additional nodes between the outer segments and the center of large deployment areas.

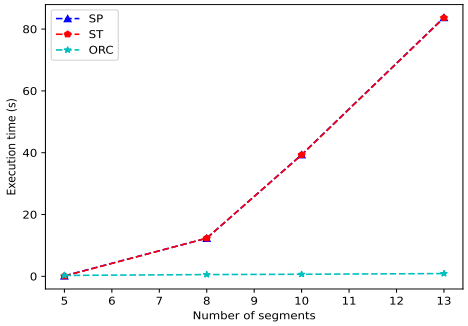
Execution time: Both ST and ORC are based on the metric closure of the graph of communication between all deployment zones to run the Steiner tree algorithm.



(a) Execution time vs number of segments ($80 \times 80m^2$)



(b) Execution time vs number of segments ($100 \times 100m^2$)



(c) Execution time vs number of segments ($120 \times 120m^2$)

Figure 4.9: Execution time as a function of segments number and deployment area size

The computation of metric closure is a time-consuming process, particularly for large graphs. Since the metric closure is computed only once for each target area, we decided to not include its calculation time in the execution time curves of Fig 4.9a, Fig 4.9b, and Fig

4.9c to avoid biasing the comparison. According to the figures, SP represents the fastest heuristic for restoring network connectivity. Since both ST and SP are computing the short paths between segments, their execution times are almost identical after computing the metric closure. Moreover, ORC has a shorter execution time than ST. This can be justified by the fact that ORC considers only three terminals at a time, whereas ST computes the Steiner tree covering all the terminals in a single run, which is more complex. In these scenarios, we have considered a large number of nearby deployment zones in a target area without obstacles. This results in a substantial communication graph with numerous edges, which slows down the tested algorithms. The objective is to assess the performance of these algorithms in selecting the minimal number of deployment zones from a vast pool of candidates. However, in real-world scenarios, these algorithms are significantly faster because the number of deployment zones is considerably smaller, and the communication graph is much less complex due to the presence of obstacles in the target area.

4.5 Conclusion

In this chapter, we present our target area modeling component based on BIM database. We explain its subcomponents, which are utilized to extract building data and model it in an exploitable format for the optimization modules. Additionally, we introduce the solution encoding, WSN representation, and the objective functions considered in both mono-objective and multi-objective cases. Moreover, we present our proposed solution to address the connectivity constraint. We provide the problem formulation, explain the introduced heuristics, and discuss the obtained results, comparing them to an existing method in the literature.

CHAPTER 5

MONO-OBJECTIVE APPROACHES FOR WSN DEPLOYMENT IN SMART BUILDINGS

5.1 Introduction

As mentioned in the previous chapter, solving the problem of WSN deployment consists of finding the optimal position of each sensor within the target area while ensuring a good trade-off between network coverage and cost. A fully connected network is also necessary in order to avoid losing data collected by isolated nodes. The WSN deployment problem has been proven to be NP-hard combinatorial optimization problem [4]. Its complexity may increase significantly due to a variety of factors, such as the type of deployment area (indoor/outdoor), its dimension model (2D/3D), obstacle heterogeneity, sensor heterogeneity, as well as the number of objective functions to be optimized [5]. Therefore, applying exact methods to multi-objective WSN deployment problem is a time-consuming task with huge complexity, especially for large and medium-sized deployment areas [5]. To cope with this problem, many researchers have applied metaheuristic algorithms to find near-optimal network topologies in a reasonable amount of time [84, 119, 69]. The decision to opt for these methods is driven by numerous benefits, including reduced computational complexity, the ability to address complex real-world problems, scalability, and the capability to integrate problem-specific heuristics to guide the search process.
In this chapter, we start by studying the behavior and compare the performance of well-known metaheuristics in solving the problem of WSN indoor deployment under real-world scenarios. The results from this evaluation were then leveraged to formulate a

hybrid mono-objective approach to further enhance the quality of the generated topologies.

5.2 Mono-objective metaheuristics for WSN deployment

5.2.1 Selection of mono-objective metaheuristic algorithms

In this section, we relied on the study presented in [5] and [6] to select the most popular algorithms used to solve the WSN deployment problem including genetic algorithm [7] (GA), differential evolution [8] (DE), cuckoo search algorithm [9] (CS), particle swarm optimization [10] (PSO), grey wolf optimizer [11] (GWO), and bat algorithm [12] (BA). Harmony search [13] (HS) is also selected since it has been successfully applied to solve the same problem [169]. The experimental tests are conducted on ten indoor architectures with different areas to evaluate also the performance and scalability of the algorithms.

In what follows, we will give a brief technical description of the selected metaheuristics and how they were applied to solve the combinatorial problem of WSN indoor deployment.

1. **Genetic Algorithm:** GA is an evolutionary algorithm designed to mimic the process of natural selection [7]. To solve the WSN deployment problem, the tested GA starts with a random population of binary vectors representing WSN topologies. Each vector has a length equal to the number of available deployment zones as explained in Section 4.3. Then, the tournament selection, the uniform crossover, and the flip multipoints mutation are used as genetic operations and applied to the population with the aim of producing new and improved solutions. This process is repeated until the maximum number of iterations is reached. The GA parameters were set according to [84].

2. **Harmony Search:** The Harmony Search (HS) algorithm is a music-inspired metaheuristic [13]. In this study, each harmony is a continuous vector representing a potential WSN topology as explained in Section 4.3. The value of each harmony component is fixed between 0 and 1. If the value is more or equal to 0.5, then a sensor will be placed at the *ith* deployment zone. The harmonies are updated according to the equations presented in the original paper [13]. The parameters are set according to [169].

3. **Particle Swarm Optimization** PSO is a stochastic method that simulates the behavior of birds flocking and fish schooling when seeking the optimal solution in the search

space [10]. The tested PSO is implemented according to the original global best topology presented in [10] yet, the inertia weight is updated according to the following equation $w_i = (Iter_{max} - Iter_i)/Iter_{max} \times (w_{max} - w_{min}) + w_{min}$ where $Iter_{max}$ and $Iter_i$ are the maximum number of iterations and current iteration, respectively, and w_{max} and w_{min} are the maximum and the minimum values of inertia weight. Each particle represents a network topology and it is encoded by a continuous vector as explained in Section 4.3. The PSO parameters were set according to [57].

4. **Differential Evolution** DE is a population-based optimization algorithm that is inspired by the natural evolution process [8]. As opposed to evolutionary algorithms, DE generates offspring by mutating a solution with a scaled difference between two randomly selected individuals. For this comparative study, the original version of DE was implemented and the DE/current-to-rand/1/bin mutation strategy has been used to generate new topologies. Parameters were set according to [170], yet the mutation factor is fixed to 0.7.

5. **Grey Wolf Optimizer** GWO is a population-based stochastic algorithm [11] which is derived from the natural predation process and hierarchy of leadership of gray wolves. A grey wolf pack has four types of leadership: alpha, beta, delta, and omega, with a decreasing dominance from alpha to omega. In the developed GWO, alpha, beta, and delta represent the best three topologies. The rest of the topologies in the population are updated in each iteration according to the hunting equation where the prey vector depends on α, β, and δ found so far as explained in [11].

6. **Bat Algorithm** BA is a swarm intelligence-based algorithm that mimics the echolocation behavior of bats [12]. In the context of this study, each bat has a position that represents the WSN deployment within the target area and a velocity vector that directs the movement of the bat in the research space. The velocity vector is updated at each iteration according to the current global best topology and the current bat frequency. The local search of the implemented BA is done around the best global topology. All the steps of BA are implemented according to [12]. The BA parameters were set according to [171].

7. **Cuckoo search algorithm** CS is a nature-inspired algorithm that emulates the brood parasitism behavior of cuckoos [9]. For this study, each cuckoo egg is a continuous vector with a length equal to the number of available deployment zones. The discretization of the cuckoo egg represents a potential sensor deployment as described in Section 4.3. Each generation creates new topologies by applying Lévy flight steps to the current

solutions as explained in [9]. Next, the new solution is compared with a randomly chosen solution from the current population, and the fittest one is maintained for the next generation. Also, the exploration of the research space is ensured by replacing a fraction p_a of worse solutions with new solutions. The CS parameters are set according to [91].

Some selected metaheuristics are designed to solve continuous problems and cannot handle the WSN deployment problem since it has been formulated as a combinatorial optimization problem. Therefore, for this study, we have adopted a simple and low-cost discretization method based on rounding decision variables as depicted in Eq. 5.1. The discretization of each solution is applied at each iteration, followed by the generation of its corresponding communication graph to repair the connectivity in case the topology is unconnected.

$$x_i^* = \begin{cases} 1, & \text{if } x_i \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

5.2.2 Performance evaluation and Comparative Study

The purpose of this section is to present the performance of GA, DE, HS, CS, GWO, BA, and PSO in solving the multi-objective WSN deployment in an indoor environment. All the algorithms were implemented in Python 3.7.16.

5.2.2.1 Experimental setup

Table 5.1 summarizes the setting parameters of the selected algorithms, derived from values suggested in studies that have tackled the WSN deployment problem with these algorithms. All tests are conducted with a population size of 100 and an iteration number of 350. A total of 20 independent runs have been conducted for each algorithm.

Table 5.5 shows the best, worst, mean, median, and standard deviation fitness values researched by each algorithm over the independent runs. Important values are highlighted in bold. The characteristics of considered sensors are summarized in Table 5.2.

Table 5.1: Simulation parameters

Algorithm	Parameter	Value
GA	Crossover probability	0.95
	Mutation Probability	0.1
	Selection technique	Tournament selection
	Tournament size	2
PSO	c1 and c2	2
	Inetria weight	decreases from 0.9 to 0.4
CS	p_a	0.3
HS	Pitch Adjustment Rate	0.3
	HMCR	0.9
DE	F	0.7
	CR	0.9
	Mutation strategy	DE/current-to-rand/1/bin
BA	Loudness	0.5
	Pulse rate	0.5
	F_{min}	0
	F_{max}	2
GWO	a parameter	$2 - 2 \times iter/Max_{iter}$

Table 5.2: Sensor physical characteristics

Parameter	Numerical value
Transmission power	3 dBm
Sensitivity	-96 dBm
Frequency	2.4 Ghz
Type of antenna	Omnidirectional
R_c	10 m
R_s	6 m
R_u	4 m
λ	0.01
γ	1

5.2.2.2 Indoor architectures for simulation tests

Due to the lack of comprehensive IFC files online detailing buildings of various area sizes, and with the aim of conducting a complete comparison, we opted to create ten indoor architectural designs, illustrated in Fig. 5.1. The proposed architectures have different areas (small, medium, and big) in order to test the scalability of the selected algorithms. Walls of the interior are constructed from plasterboard and those of the exterior are constructed from concrete.

Table 5.3: Characteristics of the proposed indoor architectures

	Area(m^2)	Number of obstacles	Number of target points
Archi1	70	14	29
Archi2	1900	30	665
Archi3	270	24	94
Archi4	180	20	62
Archi5	520	30	182
Archi6	328	18	114
Archi7	415	20	145
Archi8	700	27	280
Archi9	255	24	80
Archi10	1600	39	560

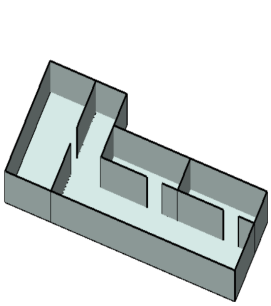
The attenuation value of each construction material are the same in [172] and depicted in Table 5.4. Each indoor architecture has a number of randomly distributed target points to be covered. Table 5.3 summarizes the characteristics of the proposed architectures.

Table 5.4: Walls attenuation

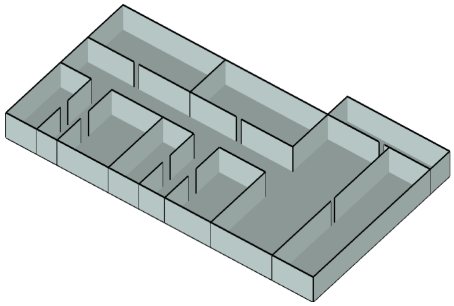
Parameter	Numerical value
Outer wall attenuation(concrete)	25 dB
Inner wall attenuation(plasterboard)	6 dB
Path loss exponent	2.8

5.2.2.3 Results and discussion

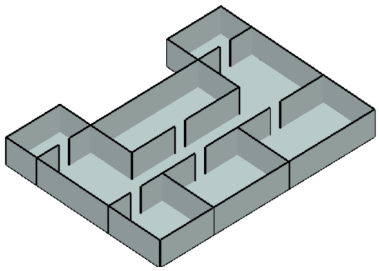
For these simulation tests, we have assumed that the deployment points where sensors could be placed are distributed on the ceilings. This reduces the complexity of the area since we are considering a 2D plane deployment zone. As can be observed from Table 5.5, DE, GA, HS, and GWO have a close performance on archi1 in terms of best fitness. DE outperforms all the algorithms on archi4 and archi6 in terms of best, mean, and median. CS and BA have the worst fitness values for small areas (archi1 and archi4). HS performs well on small and medium-sized areas. It reached the best fitness value on archi1 and the second minimal worst solution in archi4 with GA. For medium-sized areas, we notice that GWO has the best mean and median values for archi7, followed by GA. The worst values reached on medium-size architectures are generated by CS and BA. Regarding big-sized areas, we can see that the performance of DE and HS have significantly deteriorated. On the other hand, GA continues to produce good solutions with the best std value for archi2 and archi10, while GWO achieved the best results for huge zones.



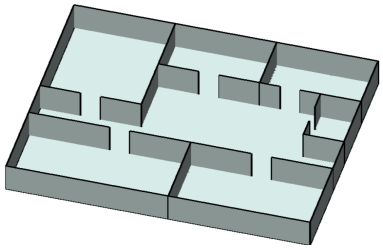
(a) Architecture 1 (70 m^2)



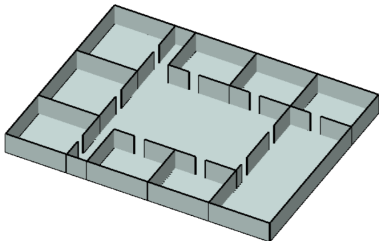
(b) Architecture 2 (1900 m^2)



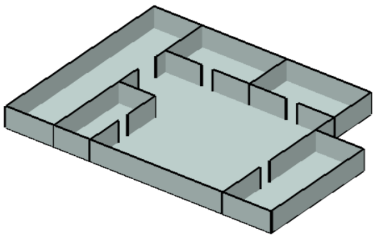
(c) Architecture 3 (270 m^2)



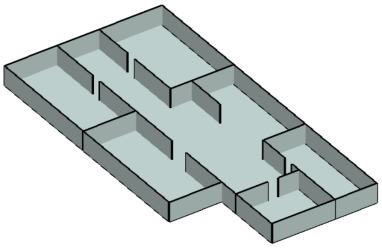
(d) Architecture 4 (180 m^2)



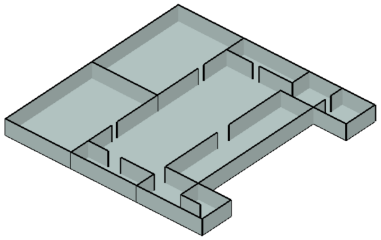
(e) Architecture 5 (520 m^2)



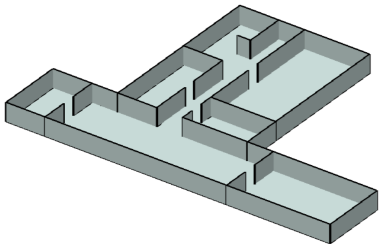
(f) Architecture 6 (328 m^2)



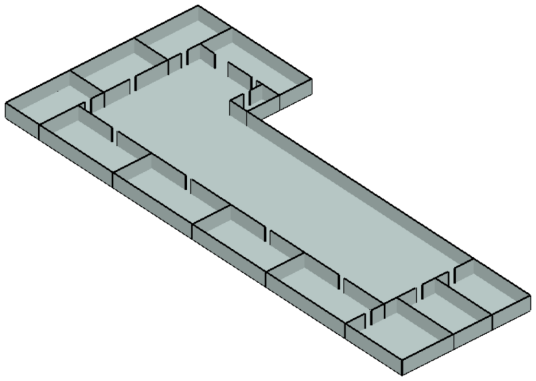
(g) Architecture 7 (415 m^2)



(h) Architecture 8 (700 m^2)



(i) Architecture 9 (255 m^2)



(j) Architecture 10 (1600 m^2)

Figure 5.1: The indoor architectures proposed for the comparison tests.

Table 5.5: Experimental results of Best, Worst, Mean, Median, and Standard Deviation (STD) of the fitness values

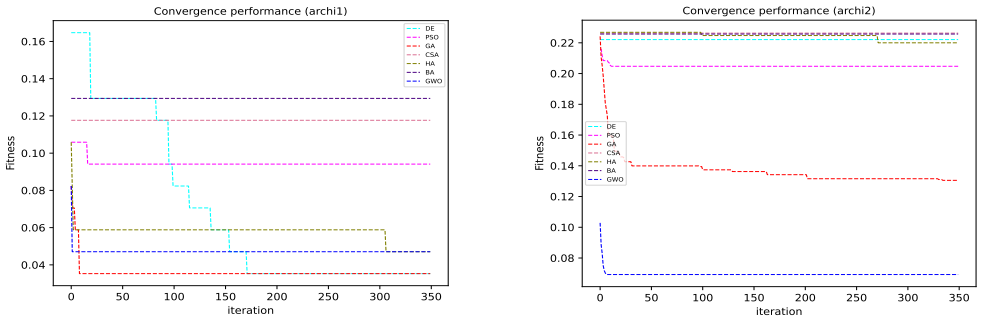
		A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
DE	Best	**0.03529**	0.21421	**0.04074**	**0.02777**	0.1	**0.04268**	0.07952	**0.14214**	**0.03529**	0.215
	Worst	0.04705	0.23632	**0.12222**	**0.06111**	0.21346	0.11280	0.19759	0.22929	0.10196	0.22902
	Mean	0.03705	0.22589	**0.05740**	**0.04027**	0.17048	0.08229	0.12458	0.19864	0.05471	0.2241
	Median	0.03529	0.22658	**0.05370**	**0.03611**	0.18076	0.08384	0.11807	0.21107	0.05098	0.22437
	STD	0.00430	0.00467	**0.01763**	**0.01123**	0.03312	0.01944	0.03643	0.0271	0.0169	0.004
PSO	Best	0.05882	0.20000	0.12593	0.09444	0.16731	0.13110	0.15422	0.18143	0.13333	0.20438
	Worst	0.12941	0.22421	0.17407	0.16667	0.19808	0.17988	0.19036	0.21143	0.18431	0.21750
	Mean	0.09467	0.21355	0.15204	0.13444	0.17923	0.16143	0.17337	0.20004	0.15373	0.21081
	Median	0.09412	0.21395	0.15556	0.13889	0.17885	0.16159	0.17108	0.20143	0.14902	0.21063
	STD	0.01996	0.00533	0.01459	0.01781	0.00850	0.01288	0.01070	0.00873	0.01597	0.00353
GA	Best	**0.03529**	**0.12722**	0.12592	0.03333	**0.06730**	**0.04573**	0.06506	**0.09143**	**0.03922**	**0.12187**
	Worst	0.03529	**0.13526**	0.19259	0.04444	**0.08076**	**0.06707**	0.07470	0.10143	**0.0549**	**0.12795**
	Mean	0.03529	**0.13209**	0.15777	0.04108	**0.07678**	**0.06061**	0.07062	0.09604	0.04824	**0.12519**
	Median	0.03529	**0.13256**	0.16111	0.03888	**0.07692**	**0.06097**	0.07196	0.09607	0.04706	**0.12558**
	STD	0.0	**0.00228**	0.01679	0.00329	**0.00302**	**0.00407**	0.00289	0.0031	0.00339	**0.00194**
CS	Best	0.10588	0.21895	0.17037	0.14444	0.18654	0.17378	0.16627	0.19929	0.17255	0.20938
	Worst	0.16471	0.23053	0.20000	0.18889	0.21923	0.20427	0.20964	0.22786	0.20392	0.23250
	Mean	0.13706	0.22605	0.18738	0.17220	0.20346	0.19099	0.19771	0.21718	0.18906	0.22381
	Median	0.13529	0.22684	0.18889	0.17222	0.20385	0.19207	0.20000	0.21857	0.19020	0.22469
	STD	0.01632	0.00356	0.00722	0.01373	0.00699	0.00804	0.01115	0.00834	0.01009	0.00558
HS	Best	**0.03529**	0.21789	0.08889	0.03889	0.15769	0.10976	0.13976	0.19714	0.07843	0.21125
	Worst	0.05882	0.23579	0.10741	0.06667	0.18846	0.13415	0.16627	0.22714	0.10588	0.23062
	Mean	0.04118	0.22576	0.09722	0.05528	0.17749	0.12149	0.15217	0.21050	0.09176	0.22119
	Median	0.03529	0.22632	0.09815	0.05556	0.17981	0.12195	0.14940	0.21000	0.09020	0.22250
	STD	0.00714	0.00459	0.00549	0.00686	0.00836	0.00658	0.00825	0.00676	0.00768	0.00587
BA	Best	0.11765	0.21789	0.16667	0.14444	0.19038	0.16159	0.18072	0.21357	0.16078	0.21000
	Worst	0.17647	0.23632	0.20000	0.18889	0.21538	0.20427	0.20964	0.23143	0.20392	0.23062
	Mean	0.14290	0.22650	0.18738	0.17106	0.20538	0.18962	0.19843	0.22182	0.18455	0.22268
	Median	0.14118	0.22526	0.18889	0.16944	0.20577	0.19055	0.19880	0.22214	0.18468	0.22281
	STD	0.01635	0.00407	0.01070	0.01219	0.00718	0.00977	0.00939	0.00489	0.01349	0.00519
GWO	Best	**0.03529**	**0.06835**	**0.04074**	0.03888	**0.05**	**0.04573**	**0.04337**	**0.07286**	**0.03137**	**0.06714**
	Worst	0.07058	**0.08805**	**0.06619**	0.08225	**0.09972**	0.07852	**0.08367**	**0.105**	0.07132	**0.08741**
	Mean	0.05	**0.07532**	**0.05379**	0.05370	**0.06278**	0.06045	**0.05942**	**0.08736**	0.05324	**0.07702**
	Median	0.04705	**0.07410**	**0.05346**	0.05250	**0.06098**	0.0606	**0.05783**	**0.08750**	0.05564	**0.07683**
	STD	0.00842	**0.00502**	**0.00724**	0.01077	**0.01210**	0.00785	**0.00953**	**0.00845**	0.00984	**0.00517**

Thus, in summary, we can say that DE, GA, HS, and GWO are efficient in small and medium-sized architectures. Moreover, GWO and GA are both scalable due to their ability to maintain their performance for large areas. Conversely, PSO, CS, and BA have the worst values for all architectures. From the obtained results, it can be concluded that the search strategy of GWO is very efficient when dealing with the WSN deployment problem. It allows to update the distribution of sensor nodes in each topology according to the three best WSN topologies found so far with a random shift caused by the control vectors. This balance between the exploration and exploitation in the search strategy enables GWO to handle high dimensional WSN deployment problem (archi2 and archi10). Furthermore, the search technique of GA has also demonstrated its effectiveness in addressing the WSN deployment problem. This is because of its suitability for combinatorial problems and its ability to maintain population diversity, particularly when using the uniform crossover. In the context of WSN deployment, DE performs well on small and medium target areas, yet its mutation technique fails to produce good topologies when the area gets bigger.

All the algorithms are compared statistically using the nonparametric test Kruskal-Wallis with significance level of 0.05, followed by the Tukey HSD. Results are reported in Table 5.6. All the *p*-values that are below 0.05 indicate a significant difference in the performance of the two examined algorithms. Fig. 5.2 depicts the convergence curves of all algorithms for one dependent run. The first iteration corresponds to the first updated population. According to it, PSO, CS, and BA got stagnated in the early stages of algorithms and this is for almost all architectures. DE and HS have both a good convergence for small and medium-sized architectures. Yet, both algorithms get stagnated for archi2 and archi10 (big-sized architectures).

Table 5.6: p-values produced by Kruskal-Wallis test followed by the Tukey pairwise comparison

	Archi1	Archi2	Archi3	Archi4	Archi5	Archi6	Archi7	Archi8	Archi9	Archi10
(GWO, GA)	5.75e-08 0.004	6.23e-08 -0.0	0.23421 0.99	0.029 0.99	4.90e-06 0.0455	0.56 1.0	5.03e-05 0.3048	0.0002 0.2706	0.017 0.8375	6.15e-08 -0.0
(GWO, HS)	0.001 0.27	6.24e-08 -0.0	5.58e-08 -0.0	0.09 0.99	6.06e-08 -0.0	5.86e-08 -0.0	5.91e-8 -0.0	6.16e-08 -0.0	5.53e-08 -0.0	6.19e-08 -0.0
(GWO, DE)	2.64e-06 0.02	6.27e-08 -0.0	0.88 0.9321	6.06e-08 0.006	6.23e-08 -0.0	0.0001 0.0	8.31e-8 -0.0	6.23e-08 -0.0	0.65 0.9997	6.15e-08 -0.0
(GWO, PSO)	1.22e-07 -0.0	6.27e-08 -0.0	5.9e-08 -0.0	0.002 -0.0	6.16e-08 -0.0	6.23e-08 -0.0	5.95e-08 -0.0	6.19e-08 -0.0	5.87e-08 -0.0	6.16e-08 -0.0
(GWO, CS)	3.67e-08 -0.0	6.23e-08 -0.0	5.56e-08 -0.0	5.63e-08 -0.0	6.01e-08 -0.0	6.0e-08 -0.0	5.96e-8 -0.0	6.19e-08 -0.0	5.85e-08 -0.0	6.18e-08 -0.0
(GWO, BA)	3.67e-08 -0.0	6.09e-08 -0.0	5.83e-08 -0.0	5.56e-08 -0.0	6.16e-08 -0.0	5.95e-08 -0.0	6.01e-08 -0.0	6.21e-08 -0.0	5.97e-08 -0.0	6.25e-08 -0.0
(GA, DE)	0.075 0.99	6.20e-08 -0.0	0.29 0.99	5.79e-08 0.001	5.88e-08 -0.0	0.0001 0.0	5.96e-8 -0.0	6.13e-08 -0.0	0.29 0.6047	6.038e-08 -0.0
(GA, HS)	0.0008 0.74	6.17e-08 -0.0	3.73e-08 -0.0	2.9e-05 1.0	5.71e-08 0.7351	5.39e-08 -0.0	5.72e-8 -0.0	6.06e-08 -0.0	3.27e-08 -0.0	6.08e-08 -0.0
(GA, PSO)	6.86e-9 -0.0	6.2e-08 -0.0	3.96e-08 -0.0	0.99 -0.0	5.81e-08 -0.0	5.47e-08 -0.0	5.91e-08 -0.0	6.09e-08 -0.0	3.49e-08 -0.0	6.04e-08 -0.0
(GA, BA)	6.56e-9 -0.0	6.02e-08 -0.0	3.91e-08 -0.0	3.83e-08 -0.0	5.81e-08 -0.0	5.47e-08 -0.0	5.82e-08 -0.0	6.11e-08 -0.0	3.55e-08 -0.0	6.14e-08 -0.0
(GA, CS)	6.57e-09 -0.0	6.16e-08 -0.0	3.72e-08 -0.0	3.94e-08 -0.0	5.6704e-8 -0.0	5.52e-08 -0.0	5.77e-08 -0.0	6.09e-08 -0.0	3.47e-08 -0.0	6.07e-08 -0.0
(DE, CS)	1.62e-08 -0.0	0.86 1.0	5.57e-08 -0.0	3.78e-08 -0.0	0.0003 0.0	6.06e-08 -0.0	4.34e-07 -0.0	0.034 0.0001	5.91e-08 -0.0	0.98 1.0
(HS, BA)	3.68e-8 -0.0	0.57 -0.0048	5.59e-08 -0.0	5.48e-08 -0.0	5.99e-08 0.0	5.76e-08 -0.0	5.83e-08 0.0	7.44e-06 0.0563	5.59e-08 -0.0	0.57 0.9494
(PSO, BA)	0.7489 -0.0	1.76e-7 -0.017	3.12e-07 -0.0	6.759e-07 -0.0	1.7224e-07 0.0	3.38e-7 0.0	1.22e-06 0.0001	6.21e-08 0.0	4.51e-06 0.0	6.31e-07 0.0



(a) (b)

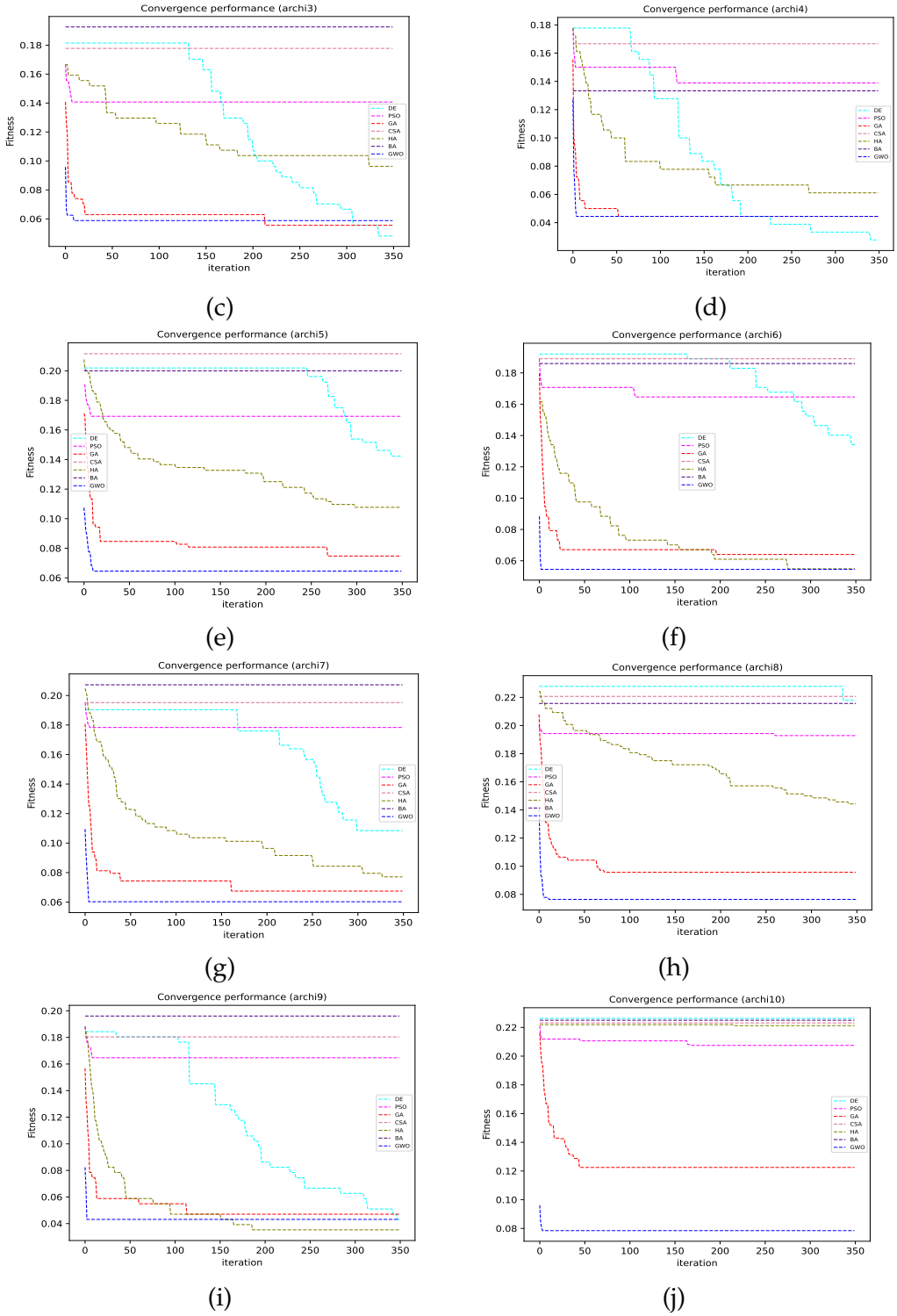


Figure 5.2: Convergence graphs of one independent run for: (a) archi1, (b) archi2, (c) archi3, (d) archi4, (e) archi5, (f) archi6, (g) archi7, (h) archi8, (i) archi9, (j) archi10

GWO generates good solutions from the first update of the population and has fast con-

vergence even for large areas, maintaining a significant difference between the second-best algorithm (GA).

5.3 A hybrid binary grey wolf optimizer for WSN deployment

The obtained results from the comparative study validated the efficacy and scalability of the GWO in comparison to the other algorithms. Therefore, this section introduces a novel approach based on GWO for addressing the problem of WSN deployment in indoor environments.

5.3.1 An overview on GWO

GWO was introduced by Mirjalili et al [11]. It is an optimization algorithm designed to mimic the hierarchical structure and predatory behavior observed in grey wolves in the wild. The leadership hierarchy of the wolf pack comprises four types of wolves: alpha, beta, delta, and omega, the dominance gradually decreases from alpha to omega. In order to establish a mathematical representation, the GWO algorithm assigns the fittest solution as the α grey wolf, the second-best solution as the β grey wolf, the third-best solution as the δ grey wolf, and considers the remaining population to be made up of ω grey wolves. During the optimization process, the GWO algorithm employs three hunting techniques: encircling prey, hunting, and attacking prey. These techniques involve updating the positions of the ω wolves based on α, β, and δ positions to guide the search process in looking for the global best solution.

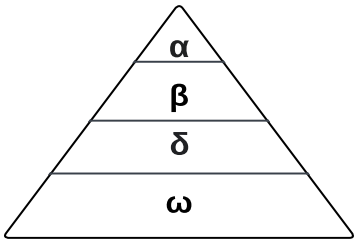


Figure 5.3: The hierarchical structure of grey wolves [11]

Encircling prey: the equation of encircling prey is as follows:

$$\vec{D} = |\vec{C}.\vec{X}_p(t) - \vec{X}(t)| \tag{5.2}$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A}.\vec{D} \tag{5.3}$$

where : $\vec{X}(t)$ represents the current position of the wolf, $\vec{X_p}(t)$ is the position of the prey, $\vec{A}$ and $\vec{C}$ are coefficient vectors calculated as in Eq.5.4 and Eq.5.5 respectively.

$$\vec{A} = 2.\vec{a}.\vec{r_1} - \vec{a} \tag{5.4}$$

$$\vec{C} = 2.\vec{r_2} \tag{5.5}$$

$\vec{r_1}$ and $\vec{r_2}$ are random vectors in $[0,1]$, a decreases linearly from 2 to 0 as follows :

$$a = 2 - 2 \times \frac{t_i}{T_{max}} \tag{5.6}$$

t_i is the current iteration and T_{max} is the maximum number of iteration.

Hunting: the hunting process of GWO involves computing for each ω wolf, the components $\vec{X_1}$, $\vec{X_2}$, $\vec{X_3}$ of its current position $\vec{X}(t+1)$ as follows:

$$\begin{aligned} \vec{D_\alpha} &= |\vec{C_1}.\vec{X_\alpha} - \vec{X}| \\ \vec{D_\beta} &= |\vec{C_2}.\vec{X_\beta} - \vec{X}| \\ \vec{D_\delta} &= |\vec{C_3}.\vec{X_\delta} - \vec{X}| \end{aligned} \tag{5.7}$$

$$\begin{aligned} \vec{X_1} &= \vec{X_\alpha} - \vec{A_1}.\vec{D_\alpha} \\ \vec{X_2} &= \vec{X_\beta} - \vec{A_2}.\vec{D_\beta} \\ \vec{X_3} &= \vec{X_\delta} - \vec{A_3}.\vec{D_\delta} \end{aligned} \tag{5.8}$$

$$\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{5.9}$$

The pseudo-code of the GWO is depicted in algorithm 9.

Algorithm 9: Continuous Grey Wolf Optimizer

Data: Pop_{size} /* number of the wolves in the population*/
T_{max} /*maximum number of iteration*/
Result: α: /*the best grey wolf in the population*/
$Pop \leftarrow$ initial random population;
$t \leftarrow 0$;
while $t < T_{max}$ **do**
  compute a according to Eq.5.6;
  $\alpha, \beta, \delta \leftarrow$ get best solution of Pop;
  for *each* $wolf_i$ *in* Pop **do**
    compute $\vec{A}_1, \vec{A}_2, \vec{A}_3$ according to Eq.5.4;
    compute $\vec{C}_1, \vec{C}_2, \vec{C}_3$ according to Eq.5.5;
    compute $\vec{X}_1, \vec{C}_2, \vec{C}_3$ according to Eq.5.5;
    $wool f_i$.pos $\leftarrow$ update position according to Eq.5.9;
    $wool f_i$.fitness $\leftarrow$ compute fitness;
  end
end
update α;
Return α;

5.3.2 Binary GWO

The GWO is mainly designed to solve continuous problems since each grey wolf in the population is encoded as a continuous vector. Therefore, in order to address combinatorial problems such as the WSN deployment problem, the GWO should be adapted effectively to generate binary solutions.

After the wolf position update, each continuous value x_t of the position vector is discretized according to the following equation in Section 4.3.

5.3.3 Steiner tree-based heuristic for deployment cost reduction

As mentioned before, a good WSN topology should guarantee a high coverage rate with a minimal number of deployed nodes while ensuring the connectivity constraint. The task of determining which sensors can be deactivated within the network is challenging, as the deactivation of inappropriate sensors can result in network partitioning or a reduction

in coverage. To address this issue, we have developed a Steiner tree-based heuristic that aims to identify a subset of sensors within a WSN topology that should be preserved to ensure coverage of the same target points while satisfying connectivity constraints. Consequently, all sensors not included in the selected subset are eliminated from the final topology, thereby reducing the deployment cost.

The first step of the proposed heuristic consists of linking the communication graph with the coverage graph. In the coverage graph, target points and the corresponding sensors providing coverage are represented as vertices. An edge is formed between a target point and a sensor node if the sensor covers the respective target. To enable the linkage between the two graphs, we introduce the augmented communication graph. The concept involves augmenting the WSN communication graph by incorporating virtual nodes that represent targets and virtual links that correspond to the edges within the coverage graph. The process of calculating the augmented communication graph is illustrated in Algorithm 10.

Algorithm 10: Augmented communication graph

Data: WSN /*communication graph*/,
T /*list of target points*/
Result: $WSN_{augmented}$: /*augmented wsn*/
$WSN_{augmented} \leftarrow WSN$;
for *each* $t \in T$ **do**
  $sensors \leftarrow$ covered (t, $WSN.nodes$);
  if *sensors not empty* **then**
    for *each* $s \in sensors$ **do**
      $WSN_{augmented}$.add-edge (s, t);
    end
  end
end
Return $WSN_{augmented}$;

Once the augmented communication graph is generated, we proceed to implement the Steiner tree algorithm. This algorithm refers to the minimum spanning tree that links a given set of vertices in an undirected graph. This set of vertices is known as terminals. In our proposed heuristic, these terminals represent the target points that have already been covered by the WSN. By applying the Steiner tree algorithm to the augmented communication graph, we ensure that at least one sensor is retained for each target point, guaranteeing the same coverage rate as the initial WSN topology prior to the cost reduction

process. As the Steiner tree algorithm operates on the augmented communication graph, it effectively maintains connectivity among the selected sensor nodes. Consequently, any nodes that have not been chosen are subsequently eliminated from the WSN, resulting in a refined WSN topology. Algorithm 11 summarizes the main steps of the proposed heuristic for deployment cost reduction. Fig. 5.4 depicts an example of an augmented communication graph.

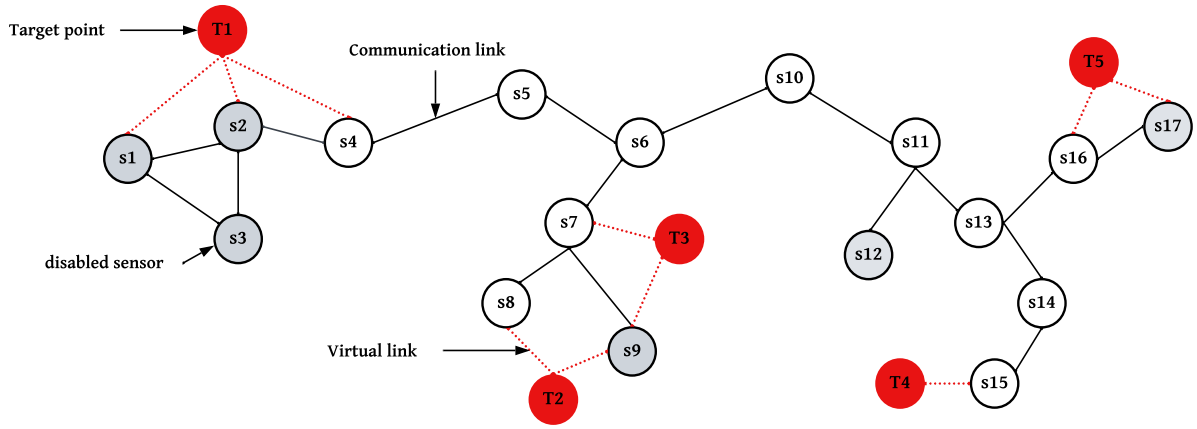


Figure 5.4: Example of augmented communication graph

The nodes s_i represent the sensor nodes of the WSN communication graph and the t_j nodes represent the added target nodes. In this example, the gray nodes denote the nodes that have been removed by the proposed heuristic, resulting in a reduction of the overall cost by six nodes.
In HBGWO, the Steiner tree-based heuristic for deployment cost reduction is applied to each topology after repairing its connectivity as depicted in Fig. 5.5.

5.3.4 Performance evaluation

The objective of this section is the test the performance of HBGWO and conduct a comparative analysis with other existing approaches in the literature. All algorithms were implemented and tested in Python 3.7.16. The characteristics of the sensors considered in this study, as well as the indoor architectures, align with those in the previous comparative study.

Algorithm 11: Steiner tree-based heuristic for deployment cost reduction

Data: *Terminals* /*list of target points covered by the nodes of WSN*/
WSN /*communication graph*/
Result: $wsn_{reduced}$: /* WSN with minimum enabled sensor nodes*/
$WSN_{augmented} \leftarrow$ augmentedCommunicationGraph (WSN);
Metric $\leftarrow$ the metric closure of $WSN_{augmented}$ /**Metric* is the complete graph induced by $WSN_{augmented}$ in which the edge weight between each pair of nodes represents their short path distance*/;
$Sub - metric \leftarrow$ the subgraph of *Metric* whose vertices represent *Terminals*;
$ST \leftarrow$ the minimum spanning tree of $Sub - metric$;
for *each* $n \in SWSN.nodes$ **do**
    if $n \notin ST.nodes$ **then**
        WSN.removeNode(n);
    end
end
Return WSN;

5.3.4.1 BIM models for simulation tests

Three BIM models have been used to assess the performance of HBGWO. Fig. 5.6 provides a visual representation of the digital models of the buildings, along with their corresponding internal layouts. To conduct the experimental tests, a total of six scenarios were derived from the BIM models. The specific characteristics of each scenario are summarized in Table 5.7. In every scenario, our focus was on deploying the WSN specifically on the first floor of the building. To simplify the target area, we limited the deployment points exclusively to the ceilings. The three BIM models used in the study had varying numbers of deployment points: BIM model 1 had 575 points, BIM model 2 had 254 points, and BIM model 3 had 257 points.

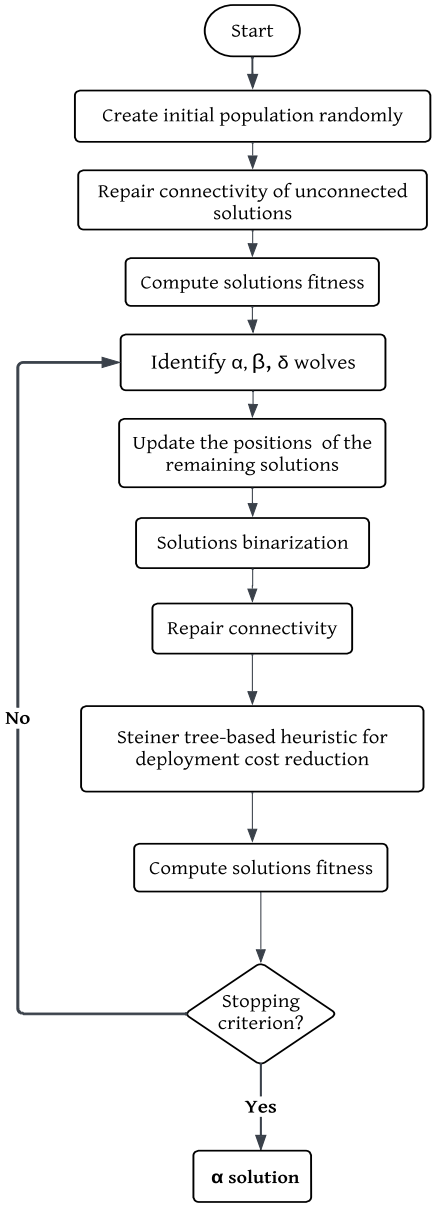
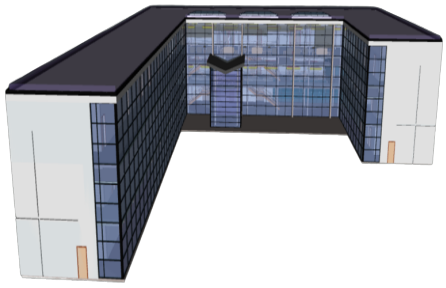


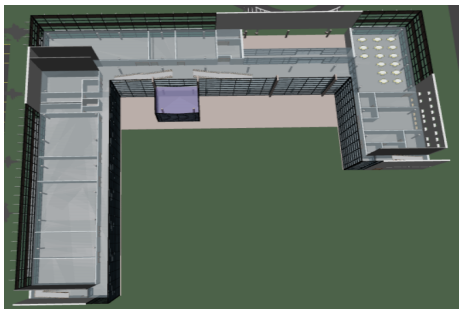
Figure 5.5: Flow chart of the proposed hybrid binary GWO

Table 5.7: Characteristics of the proposed scenarios

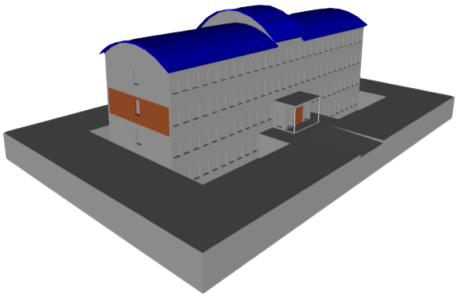
	BIM model	Number of obstacles	Number of target points
Scenario1	1	86	200
Scenario2	2	97	120
Scenario3	3	104	140
Scenario4	1	86	300
Scenario5	2	94	160
Scenario6	3	104	180



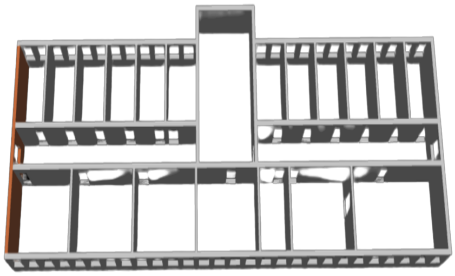
(a) BIM model 1



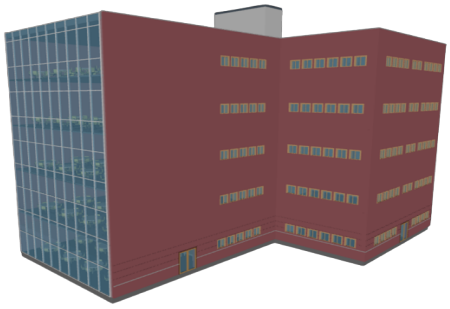
(b) Layout of BIM model 1



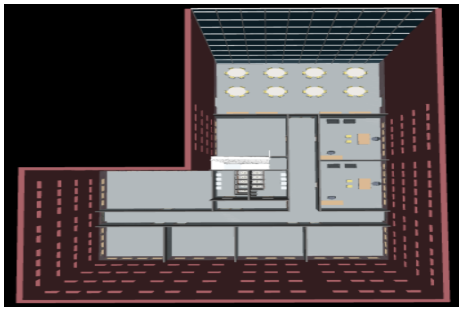
(c) BIM model 2



(d) Layout of BIM model 2



(e) BIM model 3



(f) Layout of BIM model 3

Figure 5.6: The BIM models used for the experimental tests

The attenuation characteristics of the construction materials utilized were set based on [173] and depicted in Table 5.8.

Table 5.8: Obstacles attenuation

Parameter	Numerical value
Plasterboard wall attenuation	6 dB
Glass attenuation	2.5 dB
Wood attenuation	2.7 dB
Path loss exponent	2.8

5.3.4.2 The selected algorithms for the tests

The following algorithms have been selected to compare and evaluate the performance of HBGWO.

– *BGWO*: represents the basic GWO with the binarization procedure.

– $BGWO_{v1}$: represents the binary GWO with the nonlinear convergence factor strategy defined as follows [118]:

$$a = 2 - 2 \times \sqrt{1 - (\frac{T_{max} - t_i}{T_{max}})^2}$$

– $BGWO_{v2}$: represents the binary GWO with the dynamic variation strategy defined as follows [174, 118]:

$$\begin{cases} \text{variation}, & random < 0.5 \times \frac{T_{max} - t_i}{T_{max}} \\ \text{no variation}, & otherwise \end{cases}$$

In case of *variation*, ω wolves are mutated as follows:

$$\begin{cases} d_{upper} = l_{upper} - \alpha \\ d_{lower} = \alpha - l_{lower} \\ \omega = \alpha + random \times d_{upper}, & random > 0.5 \\ \omega = \alpha - random \times d_{lower}, & otherwise \end{cases}$$

– $BGWO_{v3}$: represents the binary GWO where the ω wolves positions are updated according to the dynamic weighting strategy defined as follows [175], [118]:

$$\begin{cases} w_1 = f_\alpha / (f_\alpha + f_\beta + f_\gamma) \\ w_2 = f_\beta / (f_\alpha + f_\beta + f_\gamma) \\ w_3 = f_\gamma / (f_\alpha + f_\beta + f_\gamma) \end{cases}$$

and the updated position of ω is as follows:

$\vec{X}(t+1) = w_1 \times \vec{X_1} + w_2 \times \vec{X_2} + w_3 \times \vec{X_3}$

– *IBGWO*: the improved GWO introduced in [118].

5.3.4.3 Results and discussion

Table 5.10 depicts the experimental results of best, worst, mean, median, and standard deviation (STD) of the fitness values of all algorithms. As can be noticed from Table 5.10, the HBGWO algorithm exhibits superior performance in terms of the fitness function across all scenarios. Notably, the median value achieved by HBGWO for scenarios 1, 2, 3, and 4 is lower than the best fitness achieved by the BGWO, BGWOv1, BGWOv2, BGWOv3, and IBGWO algorithms. In scenarios 5 and 6, HBGWO surpasses the four improved algorithms when comparing the median value of HBGWO to their respective best fitness values. This difference is due to the significant decrease in deployment cost achieved through the application of the Steiner tree-based heuristic. Statistical comparison of all algorithms was conducted using the nonparametric Kruskal-Wallis test with a significance level of 0.05 followed by pairwise Tukey HSD test. The results, presented in Table 5.9 and Table 5.11, reveal that p-values below 0.05 indicate a significant distinction in the performance between the two examined algorithms. Fig. 5.7 illustrates the convergence graph of all algorithms for the median run. Analysis of this graph reveals that the HBGWO algorithm has superior performance from the initial iterations onwards, consistently maintaining lower fitness values compared to the other algorithms across all scenarios. It's worth noting that all algorithms have nearly identical coverage rates for all scenarios, varying between 89% and 98% depending on the size of the target area. However, the gap between the fitness values is related to the difference in deployment cost. Both HBGWO and BGWO demonstrate a higher capability to avoid local optima in comparison to the other improved algorithms, as they prioritize the exploration process. Conversely, IBGWO displays the poorest performance for all scenarios.

Table 5.9: Kruskal wallis test

	H statistics	p-value
Scenario1	155.42	9.3545e-32
Scenario2	152.27	4.392e-31
Scenario3	148.98	2.1956e-30
Scenario4	148.79	2.4175e-30
Scenario5	146.18	8.6738e-30
Scenario6	139.49	2.2977e-28

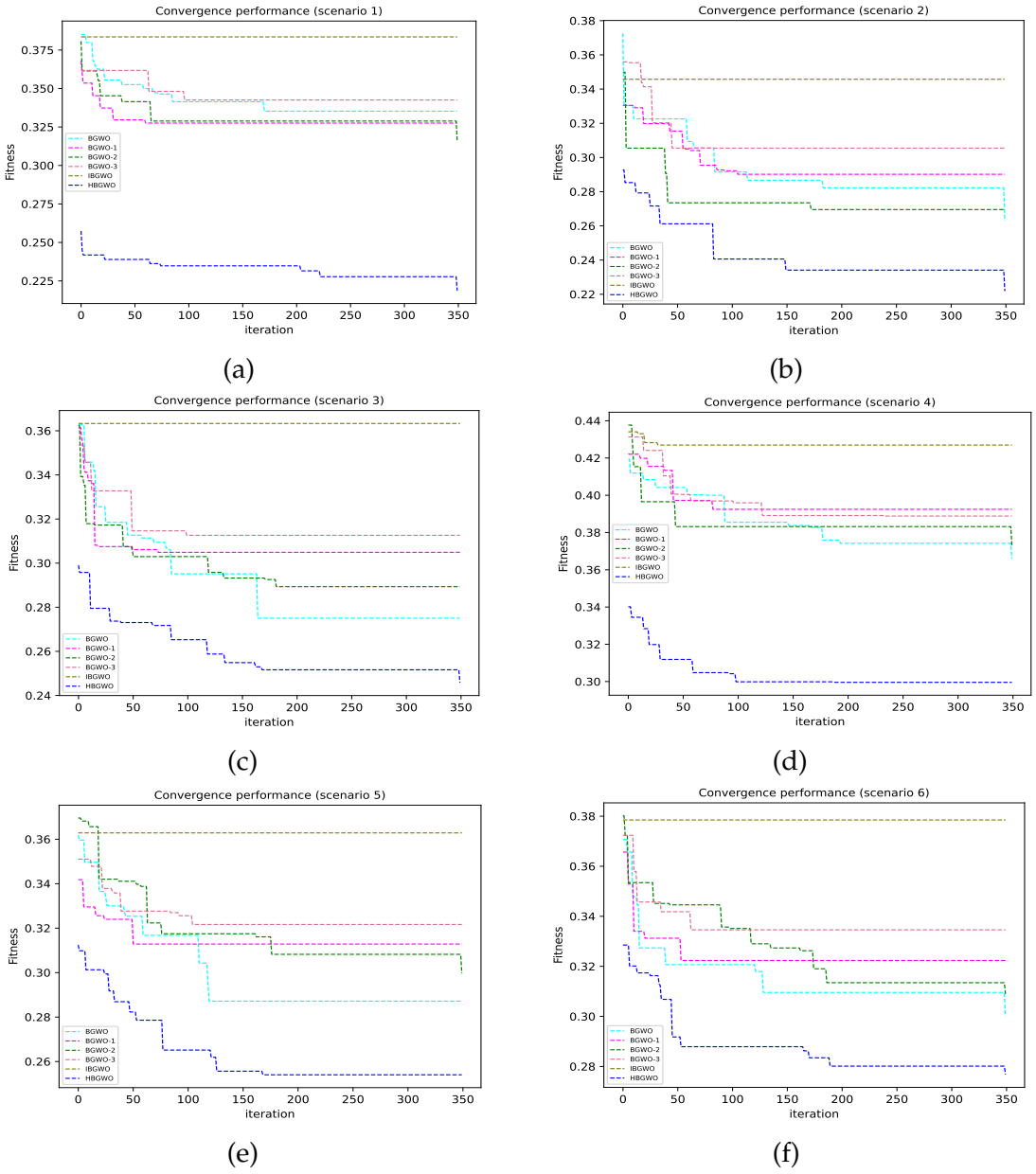


Figure 5.7: Convergence graphs of the median run for: (a) scenario 1, (b) scenario 2, (c) scenario 3, (d) scenario 4, (e) scenario 5, (f) scenario 6

Table 5.10: Experimental results of Best, Worst, Mean, Median, and Standard Deviation (STD) of the fitness values

		Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6
	Best	0.26035	0.23064	0.24972	0.33841	0.25472	0.26623
	Worst	0.33768	0.29803	0.31140	0.38072	0.32116	0.33067
$BGWO$	Mean	0.30409	0.26305	0.27528	0.36681	0.28686	0.29954
	Median	0.30947	0.26457	0.27568	0.36630	0.28784	0.30096
	STD	0.02112	0.01831	0.01295	0.00822	0.01539	0.01432
	Best	0.31148	0.26909	0.28869	0.36551	0.28853	0.29460
	Worst	0.36056	0.32467	0.33730	0.41058	0.32997	0.33738
$BGWO_{v1}$	Mean	0.33072	0.29253	0.30832	0.38926	0.31320	0.32094
	Median	0.32757	0.29213	0.30553	0.39254	0.31378	0.32263
	STD	0.01191	0.01247	0.01214	0.01168	0.01188	0.01097
	Best	0.28803	0.23412	0.25361	0.34754	0.27603	0.27289
	Worst	0.33944	0.30381	0.32046	0.39841	0.33391	0.34070
$BGWO_{v2}$	Mean	0.31312	0.27085	0.28978	0.37417	0.30062	0.30741
	Median	0.31609	0.27024	0.29063	0.37406	0.30034	0.31096
	STD	0.01282	0.02084	0.01953	0.01297	0.01673	0.01518
	Best	0.32373	0.27789	0.28991	0.37261	0.30522	0.28292
	Worst	0.37972	0.34482	0.34572	0.41986	0.35359	0.37957
$BGWO_{v3}$	Mean	0.34698	0.30573	0.31229	0.39087	0.32168	0.33352
	Median	0.34342	0.30545	0.31265	0.38884	0.32281	0.33487
	STD	0.01369	0.01269	0.01228	0.01248	0.01162	0.02057
	Best	0.34648	0.30984	0.34769	0.40928	0.33046	0.34570
	Worst	0.39570	0.36634	0.38016	0.43913	0.38578	0.39347
$IBGWO$	Mean	0.38011	0.34669	0.36368	0.42607	0.36369	0.37613
	Median	0.38366	0.34711	0.36430	0.42710	0.36297	0.37903
	STD	0.01083	0.01259	0.00907	0.00792	0.01031	0.01182
	Best	**0.20106**	**0.20079**	**0.21273**	**0.27493**	**0.23066**	**0.24179**
	Worst	**0.23831**	**0.24593**	**0.27568**	**0.31768**	**0.28622**	**0.30236**
$HBGWO$	Mean	**0.21900**	**0.22226**	**0.24465**	**0.29981**	**0.25696**	**0.27687**
	Median	**0.21880**	**0.22185**	**0.24615**	**0.29971**	**0.25485**	**0.27736**
	STD	**0.00906**	**0.01224**	**0.01528**	**0.00992**	**0.01506**	**0.01470**

Table 5.11: The Tukey pairwise comparison

		Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6
($BGWO$, $BGWO_{v1}$)	Meandiff	0.0266	0.0295	0.033	0.0224	0.0263	0.0214
	p-value	0.0	0.0	-0.0	0.0	0.0	0.0
	Reject	True	True	True	True	True	True
($BGWO$, $BGWO_{v2}$)	Meandiff	0.009	0.0078	0.0145	0.0074	0.0138	0.0079
	p-value	0.119	0.3565	0.0011	0.0884	0.0019	0.3216
	Reject	False	False	True	False	True	False
($BGWO$, $BGWO_{v3}$)	Meandiff	0.0429	0.0427	0.037	0.0241	0.0348	0.034
	p-value	-0.0	-0.0	-0.0	0.0	-0.0	0.0
	Reject	True	True	True	True	True	True
($BGWO$, $IBGWO$)	Meandiff	0.076	0.0836	0.0884	0.0593	0.0768	0.0766
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($BGWO$, $HBGWO$)	Meandiff	-0.0851	-0.0408	-0.0306	-0.067	-0.0299	-0.0227
	p-value	-0.0	-0.0	0.0	-0.0	0.0	0.0
	Reject	True	True	True	True	True	True
($BGWO_{v1}$, $BGWO_{v2}$)	Meandiff	-0.0176	-0.0217	-0.0185	-0.0151	-0.0126	-0.0135
	p-value	0.0	0.0	0.0	0.0	0.0063	0.0073
	Reject	True	True	True	True	True	True
($BGWO_{v1}$, $BGWO_{v3}$)	Meandiff	0.0163	0.0132	0.004	0.0016	0.0085	0.0126
	p-value	0.0001	0.0124	0.8791	0.9919	0.1627	0.0162
	Reject	True	True	False		False	True
($BGWO_{v1}$, $IBGWO$)	Meandiff	0.0494	0.0542	0.0554	0.0368	0.0505	0.0552
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($BGWO_{v1}$, $HBGWO$)	Meandiff	-0.1117	-0.0703	-0.0637	-0.0894	-0.0562	-0.0441
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($BGWO_{v2}$, $BGWO_{v3}$)	Meandiff	0.0339	0.0349	0.0225	0.0167	0.0211	0.0261
	p-value	-0.0	0.0	0.0	0.0	0.0	0.0
	Reject	True	True	True	True	True	True
($BGWO_{v2}$, $IBGWO$)	Meandiff	0.067	0.0758	0.0739	0.0519	0.0631	0.0687
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($BGWO_{v2}$, $HBGWO$)	meandiff	-0.0941	-0.0486	-0.0451	-0.0744	-0.0437	-0.0305
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	0.0
	reject	True	True	True	True	True	True
($BGWO_{v3}$, $IBGWO$)	Meandiff	0.0331	0.041	0.0514	0.0352	0.042	0.0426
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($BGWO_{v3}$, $HBGWO$)	Meandiff	-0.128	-0.0835	-0.0676	-0.0911	-0.0647	-0.0567
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True
($IBGWO$, $HBGWO$)	Meandiff	0.1611	0.1244	0.119	0.1263	0.1067	0.0993
	p-value	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Reject	True	True	True	True	True	True

5.4 Conclusion

In this chapter, we have examined seven metaheuristic algorithms to determine the best topology for WSN within an indoor environment. The problem was as a single discrete optimization problem and introduced a connectivity repair heuristic based on the Dijkstra short path algorithm to link all the disjoint sensor sets by deploying a minimum number of additional nodes. Ten representative indoor architectural scenarios were generated to compare the selected algorithms. The obtained results validated the efficacy and scalability of the GWO in comparison to the other algorithms. Therefore, we introduced a novel approach based on binary GWO for addressing the problem of WSN deployment in indoor environments called HBGWO. The developed HBGWO aims to optimize both network coverage and deployment cost under the connectivity constraint. It incorporates a Steiner Tree-based heuristic designed to reduce the number of deployed sensors without affecting the network coverage. Three representative BIM databases were used to generate six scenarios in order to compare the performance of HBGWO against other approaches in the literature. The experimental results demonstrated that HBGWO surpassed all other approaches, exhibiting its efficiency in computing the best WSN topologies in terms of coverage and cost across all scenarios.

CHAPTER 6

MULTI-OBJECTIVE APPROACHES FOR WSN DEPLOYMENT IN SMART BUILDINGS

6.1 Introduction

In the previous chapter, we investigated the application of mono-objective metaheuristics to address the problem of WSN deployment in indoor environments. All algorithms aim to optimize a single objective that aggregates the network coverage and deployment cost. Although these approaches have generated promising sensor topologies, they have often limitations in exploring the search space and ensuring a trade-off between conflicting solutions. This occurs because the aggregation of objective functions often causes the algorithm to converge toward a specific region in the search space, neglecting the dominance relationship between solutions and resulting in the loss of Pareto front information. In this chapter, our focus is on addressing conflicting deployment objectives simultaneously. The advantage of these methods over aggregated approaches lies in their ability to provide decision-makers with a set of Pareto optimal solutions that more comprehensively cover the search space. For this purpose, in this chapter, we propose to investigate the performance of evolutionary and swarm-based multi-objective algorithms in addressing sensor deployment, considering both coverage and deployment cost simultaneously. Subsequently, we introduce a hybrid multi-objective algorithm that combines MOGWO with NSGA-II to enhance the exploitation process. Additionally, a two-stage local search is introduced with a certain probability to further improve the quality of the produced topologies.

6.2 Multi-objective metaheuristics for WSN deployment

The commonly used evolutionary and swarm-based metaheuristics to solve the WSN deployment problem were selected for a comparative study to evaluate their performance across diverse scenarios. Subsequently, results from this assessment were used to develop a hybrid multi-objective approach that aims to enhance further the quality of generated topologies.

6.2.1 Selection of multi-objective metaheuristic algorithms

1. **NSGA-II:** NSGA-II is a multi-objective evolutionary algorithm based on Pareto dominance. It was introduced as an enhancement of the original NSGA by Deb et al [99] and structured around four fundamental concepts [176]. These concepts are (1) Non-dominating sorting: it ensures ranking individuals without any additional parameters. (2) Elitism preservation: it guarantees the retention of non-dominated solutions for the next generation. (3) The crowding distance: it measures the density around a particular individual within the objective space. (4) The crowded tournament selection strategy: it selects parents for reproduction based on their ranking and crowding distance to preserve the population diversity.

   To address the problem of WSN deployment, the NSGA-II algorithm starts by creating an initial population of random topologies encoded in a binary vector, as elaborated in Section 4.3. Subsequently, the search procedure is carried out as outlined in Algorithm 12. The generation of new topologies is achieved using uniform crossover and inversion mutation methods.

2. **SPEA-II:** SPEA-II is a multi-objective evolutionary algorithm that represents an extension of SPEA [177]. This algorithm centers around two main features: the maintenance of a fixed-size archive and the calculation of fitness values. The archive is designed to retain the best solutions found during its execution. It is managed by a dynamic control mechanism that populates it with dominated solutions when the number of non-dominated solutions is less than the predefined archive size. Conversely, when the number of non-dominated solutions exceeds the predetermined archive size, a truncation procedure is invoked to selectively remove extra individuals while preserving the critical frontier points within the Archive. Furthermore, SPEA-II introduces a novel fitness assignment technique that combines a niching

Algorithm 12: NSGA-II for WSN deployment

Data: Pop_{size} /* number of individuals in the population*/
T_{max} /*maximum number of iteration*/
P_c /* crossover probability */
P_m /* mutation probability */

Result: $font$: /* Pareto front*/

$Pop_0 \leftarrow$ generate initial population of random sensor networks topologies;
Compute coverage and cost of all topologies;
$Q_0 \leftarrow \emptyset$;
$t \leftarrow 0$;
while $t < T_{max}$ **do**
  $R_t \leftarrow Pop_t \bigcup Q_t$;
  Repair connectivity of R_t;
  Compute coverage and cost of all topologies;
  $F \leftarrow$ Fast non-dominated sort(R_t) /* F is a list of fronts */ ;
  Calculate crowding distance(F);
  $Pop_{t+1} \leftarrow \emptyset$;
  $k \leftarrow 0$;
  while $|P_{t+1} + F_k| \leq Pop_{size}$ **do**
    $Pop_{t+1} \leftarrow Pop_{t+1} \bigcup F_k$;
    $k \leftarrow k + 1$;
  end
  Sort F_k in descending order regarding crowding distance;
  $Pop_{t+1} \leftarrow Pop_{t+1} \bigcup F_k[1 : (Pop_{size} - |Pop_{t+1}|)]$;
  $Q_{t+1} \leftarrow$ Produce new sensor topologies(Pop_{t+1}, P_c, P_m);
  $t \leftarrow t + 1$;
end
Return *archive*;

mechanism based on the concept of Pareto dominance and density information. The niching mechanism of an individual i is denoted by the raw fitness $R(i)$ and determined by Eq. 6.1:

$$R(i) = \sum_{j \in p_t \cup \bar{p}_t, j \succ i} S(j) \tag{6.1}$$

where p_t and $\bar{p}_t$ are population and archive respectively, $\succ$ represents the Pareto dominance relation, and $S(j)$ is the strength of individual j computed as follows:

$$S(i) = |\{j | j \in p_t \cup \bar{p}_t \wedge i \succ j\}| \tag{6.2}$$

The density information is used to distinguish between the individuals having the same raw fitness values and it is computed as follows:

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{6.3}$$

where σ_i^k is the distance in objective space between the individual i and the k-th nearest individual of archive and population, $k = \sqrt{|p_t| + |\bar{p}_t|}$.

The fitness of an individual i is computed as depicted in Eq. 6.4

$$F(i) = R(i) + D(i) \tag{6.4}$$

Similar to NSGA-II, a WSN topology is encoded by a binary vector representing the positions of deployed sensors within the designated area. The dominance relation among these topologies is measured based on their coverage and costs, and the reproductive techniques employed for generating new deployment schemes follow the same methods used in NSGA-II. The pseudo-code of SPEA-II for solving WSN development problem is depicted in Algorithm 13.

3. **MOPSO:** PSO is a swarm intelligence algorithm that mimics the collective behavior of a flock of birds to explore the search space and find the best solution. In PSO, each particle represents a potential solution and has a current position and velocity that combines personal and global experience to compute its new position. These attributes are adjusted in each iteration according to Eq. 6.6 and Eq. 6.5 respectively.

$$v_i(t+1) = w \times v_i(t) + c_1 \times r_1 \times (P_{pbest_i} - P_i(t)) + c_2 \times r_2 \times (P_{gbest} - P_i(t)) \tag{6.5}$$

$$P_i(t+1) = P_i(t) + v_i(t+1) \tag{6.6}$$

Algorithm 13: SPEA-II for WSN deployment

Data: Pop_{size} /* number of individuals in the population*/
T_{max} /*maximum number of iteration*/
P_c /* crossover probability */
P_m /* mutation probability */
$Arch_{size}$ /* Archive size */

Result: $font$: /* Pareto front*/

$Pop_0 \leftarrow$ generate initial population of random sensor networks topologies;
$Archive_0 \leftarrow \emptyset$;
$t \leftarrow 0$;
while $t < T_{max}$ **do**
  Repair connectivity of Pop_t;
  Compute coverage and cost of all topologies;
  Compute fitness of all topologies according to Eq. 6.4;
  $Archive_{t+1} \leftarrow$ non-dominated solution of $(Pop_t \bigcup Archive_t)$;
  if $|Archive_{t+1}| < Arch_{size}$ **then**
    $R \leftarrow$ Sort $Pop_t \bigcup Archive_t$ in ascending order regarding the fitness value;
    $Archive_{t+1} \leftarrow R[|Archive_{t+1}| : Arch_{size}]$;
  end
  if $|Archive_{t+1}| > Arch_{size}$ **then**
    $Archive_{t+1} \leftarrow$ remove iteratively an individual from $Archive_{t+1}$ using the truncation procedure;
  end
  $P_{t+1} \leftarrow$ Produce new sensor topologies ($Archive_{t+1}$, P_c, P_m);
  $t \leftarrow t + 1$;
end
Return *archive*;

The PSO was modified to address multi-objective problems by incorporating an archive and a leader selection strategy. The archive has a predefined size limit and it stores non-dominated solutions discovered during the optimization process. When the number of non-dominated solutions exceeds this limit, the MOPSO employs a method to determine which solution to eliminate. It starts by partitioning the objective space into a grid of hypercubes. The probability of choosing a hypercube is directly proportional to its degree of crowding. Subsequently, a solution is randomly selected from the chosen hypercube and removed from the archive. In this study, the solutions generated by MOPSO are continuous vectors. Following each position update, all solutions are

converted into binary vectors, and the connectivity repair heuristic is applied to correct any disconnected topologies. The pseudo-code of the MOPSO for solving WSN deployment problem is depicted in Algorithm 14.

Algorithm 14: Multi-objective PSO for WSN deployment

Data: Pop_{size} /* number of the particles in the population*/
T_{max} /*maximum number of iteration*/
$nGrid$ /* number of grids per each dimension */
$nRep$ /* archive size */
c_1, c_2 /* learning factors */
Result: $font$: /* Pareto front*/
$Pop_0 \leftarrow$ generate initial population of random sensor networks topologies;
Initialize *velocity* for each *particle* in Pop_0;
Compute coverage and cost of all topologies;
Initialize *archive* with non-dominated solutions;
grid $\leftarrow$ create grid (*archive*, *nGrid*);
Find the grid index for each topology in the *archive*;
$t \leftarrow 0$;
while $t < T_{max}$ **do**
  $w_t = w_{max} - (w_{max} - w_{min}) \times (t/T_{max})$;
  Leader $\leftarrow$ Select leader from *archive*;
  for *each* $particle_i$ *in* Pop_t **do**
    $velocity_{t+1} \leftarrow$ Compute velocity (*Leader*, $particle_i$, c_1, c_2, w) according to Eq.6.5;
    $particle_i$.pos $\leftarrow$ Update particle position according to Eq.6.6;
    $particle_i$.pos $\leftarrow$ Binarize particle position according to Eq.4.1;
    $particle_i$.pos $\leftarrow$ repair position connectivity;
    $particle_i$.fitness $\leftarrow$ Compute fitness according to Eq.4.6 ;
  end
  archive $\leftarrow$ non-dominated solutions in Pop_t and *archive*;
  Find the grid index for each topology in the *archive*;
  if *length of archive* $>$ *nRep* **then**
    archive = Delete extra sensor topologies from *archive*;
    grid $\leftarrow$ Create grid (*archive*, *nGrid*);
  end
  $t \leftarrow t + 1$;
end
Return *archive*;

4. **MOGWO:** The GWO was adapted to solve multi-objective problems by incorporating the same mechanisms of MOPSO. The pseudo-code of MOGWO for solving WSN deployment problem is depicted in Algorithm 15.

Algorithm 15: Multi-objective GWO for WSN deployment

Data: Pop_{size} /* number of the wolves in the population*/
T_{max} /*maximum number of iteration*/ $nGrid$ /* number of grids per each dimension */ $nRep$ /* archive size */

Result: $font$: /* Pareto front*/

$Pop_0 \leftarrow$ generate initial population of random sensor networks topologies;
Compute coverage and cost of all topologies;
Initialize *archive* with non-dominated solutions;
$grid \leftarrow$ create grid (*archive*, *nGrid*);
Find the grid index for each topology in the *archive*;
$t \leftarrow 0$;
while $t < T_{max}$ **do**
    Compute a;
    α, β, $\delta \leftarrow$ Select leaders from *archive*;
    for *each* $wolf_i$ *in* Pop_t **do**
        Compute $\vec{D}_\alpha$, $\vec{D}_\beta$, $\vec{D}_\delta$ according to Eq.5.7;
        Compute $\vec{X}_1$, $\vec{X}_2$, $\vec{X}_3$ according to Eq.5.8;
        $wolf_i$.pos $\leftarrow$ Update wolf position according to Eq.5.9;
        $wolf_i$.pos $\leftarrow$ Binarize wolf position according to Eq.4.1;
        $wolf_i$.pos $\leftarrow$ repair position connectivity;
        $wolf_i$.fitness $\leftarrow$ Compute fitness according to Eq.4.6;
    end
    archive $\leftarrow$ non-dominated solutions in Pop_t and *archive*;
    Find the grid index for each topology in the *archive*;
    if *length of archive* $>$ *nRep* **then**
        archive = Delete extra sensor topologies from *archive*;
        grid $\leftarrow$ Create grid (*archive*, *nGrid*);
    end
    $t \leftarrow t + 1$;
end
Return *archive*;

6.2.2 Performance evaluation and comparison study

6.2.2.1 Experimental setup

All algorithms were implemented and tested in Python 3.7.16. The characteristics of the sensors considered in this study, as well as the indoor architectures, align with those in the comparative study of mono-objective algorithms.

6.2.2.2 Parameter tuning

The performance of the selected metaheuristics in addressing the WSN deployment problem relies on the selection of their control parameters. Optimal parameter selection is crucial to ensure an effective search process and prevent premature convergence. Inadequate parameter choices may lead to an unbalance between the exploitation and exploration phases, potentially causing stagnation, premature convergence, or lack of population diversity. In this study, the Irace package was used for parameter tuning of the selected algorithms [178]. The cost function employed to assess the effectiveness of a specific parameter configuration of a metaheuristic represents the average hypervolume value of the Pareto fronts obtained when employing this metaheuristic across three diverse architectures of varying sizes. The Irace parameters were maintained at the default values specified in the original implementation. Table 6.1 summarizes the range of parameters and their optimized values.

Table 6.1: Parameters values

	Parameters	Range	Tuned value
NSGA-II	Population size	70-150	133
	Crossover probability	0.7-0.9	0.8742
	Mutation probability	0.01-0.1	0.089
SPEA-II	Population size	70-150	139
	Crossover probability	0.7-0.9	0.862
	Mutation probability	0.01-0.1	0.0589
MOPSO	Population size	70-150	94
	c1	1.5-2.0	1.9089
	c2	1.5-2.0	1.926
	NoGrid	30-50	48
	Archive size	40-50	49
MOGWO	Population size	70-150	96
	NoGrid	30-50	46
	Archive size	40-50	49

6.2.2.3 Results and discussion

Each metaheuristic has been executed 25 independent runs for each architecture. To conduct a fair study, the number of fitness evaluations has been fixed for each algorithm, for less complexity we supposed that the sensors could be placed on the ceilings.

1. **Analysis of the numerical results**
   Experimental results of best, worst, mean, median and standard deviation (STD) of hypervolume and Inverted Generational Distance (IGD) of all algorithms are depicted in Table 6.2. Several observations can be deduced from this table: regarding small architectures (archi 1, archi 3, archi 4, archi 9), the best values of hypervolume were reached by NSGA-II, MOGWO, and SPEA-II, while MOPSO provides solutions of lower quality. Furthermore, the median values for these architectures closely align with the best value, indicating that the algorithms converge toward nearly identical solution sets for all independent runs. This convergence justifies the small values of STD. The optimal Pareto front used in the calculation of IGD is estimated by combining Pareto fronts from separate algorithm runs within a specific architecture. Non-dominated solutions are then chosen to form this estimated optimal Pareto front. The worst values of IGD for small architectures are obtained by MOPSO. This means that MOPSO does not share good solutions with the optimal set.
   In medium-sized architectures (archi 5, archi 6, archi 7, archi 8), NSGA-II and MOGWO consistently maintain their high performance, while SPEA-II shows a slight decline in performance compared to the smaller architecture. However, MOPSO continues to produce the worst solutions. Similar patterns are observed in large architectures (archi 2, archi 10), albeit with a small decrease in solution quality compared to medium-sized architectures. NSGA-II achieves good results due to its ability to handle combinatorial optimization problems. Additionally, the incorporation of uniform crossover and inversion mutation has bolstered the preservation of population diversity, enhancing the exploration process. SPEA-II employs the same reproduction mechanisms and maintains the non-dominated solutions found so far through its archive. However, its fitness update mechanism struggles with medium and large architectures due to its computational complexity. This process demands numerous fitness evaluations, depleting SPEA-II's evaluation budget and restricting its ability to explore and exploit the search space. In contrast, NSGA-II benefits from a simpler fitness evaluation process, conserving its budget to facilitate more extensive exploration and exploitation of the search space.

Table 6.2: Experimental results: Best, Worst, Mean, Median, and STD for hypervolume and IGD

		NSGA-II		SPEA-II		MOGWO		MOPSO	
		HV	IGD	HV	IGD	HV	IGD	HV	IGD
Archi1	Best	**0.97282**	**0.00000**	**0.97282**	**0.00000**	**0.97282**	0.05176	0.95335	0.06716
	Worst	**0.96998**	0.06718	0.95254	0.13817	0.96836	0.07142	0.91400	0.31189
	Mean	0.97240	0.00829	0.95565	0.11790	0.97167	0.05835	0.93412	0.19838
	Median	0.97282	0.00000	0.95254	0.13817	0.97201	0.05568	0.93509	0.19850
	Std	0.00074	0.01730	0.00622	0.04142	0.00125	0.00658	0.01029	0.06693
Archi 2	Best	0.94961	0.24988	0.93822	0.19665	**0.95743**	**0.02462**	0.87809	0.33257
	Worst	0.93464	0.32649	0.93229	0.23631	0.91823	**0.03537**	0.84887	0.36859
	Mean	0.94129	0.29326	0.93520	0.21537	**0.94273**	**0.02890**	0.86016	0.35269
	Median	0.94160	0.29341	0.93560	0.21737	**0.94606**	**0.02887**	0.86021	0.35491
	Std	0.00437	0.01821	0.00190	0.01058	0.01164	**0.00326**	0.00724	0.01013
Archi 3	Best	**0.97805**	**0.01716**	0.96816	0.03663	**0.97530**	**0.02128**	0.91292	0.22321
	Worst	0.96281	0.16695	0.96084	0.08201	**0.96521**	**0.03971**	0.86667	0.40227
	Mean	**0.97419**	0.06405	0.96472	0.05684	0.97100	**0.03026**	0.89316	0.30779
	Median	**0.97470**	0.05660	0.96470	0.05459	0.97100	**0.03000**	0.89413	0.31930
	Std	0.00305	0.03506	0.00174	0.01120	0.00249	**0.00401**	0.01118	0.04156
Archi 4	Best	0.97643	0.00746	**0.97670**	**0.00445**	0.97509	0.02543	0.92366	0.20780
	Worst	0.97168	0.07132	**0.97258**	**0.02161**	0.96873	0.04063	0.87142	0.37042
	Mean	**0.97534**	0.02292	0.97425	**0.01529**	0.97223	0.03441	0.90224	0.29732
	Median	**0.97554**	0.01714	0.97410	**0.01620**	0.97222	0.03519	0.90314	0.30529
	Std	0.00098	0.01493	0.00095	**0.00425**	0.00182	0.00466	0.01138	0.04583
Archi 5	Best	**0.97431**	0.04998	0.94978	0.10601	**0.97161**	**0.01717**	0.89385	0.25973
	Worst	**0.95987**	0.20493	0.93762	0.15509	**0.96520**	**0.02944**	0.86476	0.33146
	Mean	**0.96987**	0.10407	0.94328	0.13402	**0.96904**	**0.02215**	0.87532	0.30191
	Median	**0.97065**	0.09143	0.94392	0.13497	**0.96940**	**0.02209**	0.87363	0.30268
	Std	**0.00320**	0.03722	0.00308	0.01285	**0.00170**	**0.00276**	0.00809	0.01894
Archi 6	Best	**0.97705**	**0.01480**	0.96646	0.03778	0.97363	**0.01860**	0.91431	0.22235
	Worst	**0.96801**	0.13883	0.96050	0.07981	0.96708	**0.02948**	0.86246	0.33659
	Mean	**0.97325**	0.05542	0.96311	0.05944	0.97072	**0.02460**	0.88946	0.28335
	Median	**0.97410**	0.04767	0.96325	0.05848	0.97026	**0.02531**	0.88939	0.28056
	Std	0.00269	0.03235	0.00135	0.01000	0.00150	**0.00275**	0.01217	0.02538
Archi 7	Best	**0.97668**	**0.01287**	0.95335	0.08859	**0.97113**	**0.01861**	0.89853	0.23671
	Worst	**0.96465**	0.14035	0.94375	0.13295	**0.96372**	**0.02912**	0.86476	0.34791
	Mean	0.97149	0.07143	0.94892	0.10830	0.96746	0.02357	0.88302	0.29385
	Median	0.97148	0.05838	0.94907	0.10562	0.96721	0.02301	0.88244	0.29487
	Std	0.00279	0.03494	0.00226	0.01212	0.00209	0.00302	0.00895	0.02283
Archi 8	Best	**0.94953**	**0.04880**	0.90295	0.19758	**0.94388**	**0.01852**	0.85968	0.28094
	Worst	0.93039	0.23604	0.88273	0.24744	**0.93693**	**0.02912**	0.82593	0.34195
	Mean	0.93795	0.17546	0.89342	0.22523	**0.94164**	0.02470	0.84446	0.31344
	Median	0.93837	0.17961	0.89439	0.22715	**0.94195**	0.02491	0.84346	0.31881
	Std	0.00444	0.04278	0.00523	0.01357	**0.00149**	0.00209	0.00717	0.01792
Archi 9	Best	**0.97990**	0.01278	0.97701	**0.02257**	0.97348	0.02437	0.91103	0.22907
	Worst	0.96725	0.15223	**0.96892**	**0.03806**	0.96603	0.05906	0.86975	0.36339
	Mean	**0.97480**	0.07434	0.97369	**0.02985**	0.97004	0.03974	0.89539	0.29344
	Median	**0.97461**	0.08034	0.97348	**0.02907**	0.97025	0.03909	0.89289	0.30360
	Std	0.00323	0.04438	0.00210	**0.00414**	0.00163	0.00961	0.00941	0.03862
Archi 10	Best	**0.95202**	0.22019	0.94582	0.14807	**0.95726**	**0.01963**	0.87674	0.32137
	Worst	**0.93584**	0.32586	0.93574	0.22566	0.92428	**0.03715**	0.85058	0.37707
	Mean	**0.94504**	0.27671	0.93974	0.19384	0.94636	**0.02588**	0.86173	0.35714
	Median	**0.94466**	0.27894	0.93923	0.19449	0.94802	**0.02463**	0.86143	0.35980
	Std	**0.00406**	0.02512	0.00266	0.01656	0.00777	**0.00346**	0.00498	0.01035

With regard to swarm intelligence-based algorithms MOGWO and MOPSO, MOGWO has a consistent superiority over MOPSO across all architectures. This advantage primarily stems from the MOGWO position update mechanism which ensures a wide coverage of the search space. This is done by the integration of three leaders to enhance exploration while introducing slight random perturbations to effectively avoid local optima. In contrast, MOPSO faces limitations in exploration which leads to premature convergence

Table 6.3: Kruskal wallis test

	H statistics		p-value	
	HV	IGD	HV	IGD
Archi 1	1617.69	1601.78	0.0	0.0
Archi 2	820.61	1436.73	1.47e-177	3.14e-311
Archi 3	1197.13	1047.38	3.073e-259	9.44e-227
Archi 4	1447.72	1387.15	0.0	1.8e-300
Archi 5	1064.04	1307.92	2.3e-230	2.81e-283
Archi 6	1061.06	1364.03	1.02e-229	1.87e-295
Archi 7	1230.95	1327.80	1.4e-266	1.36e-287
Archi 8	1028.69	1327.80	1.07e-222	1.36e-287
Archi 9	1065.43	1055.80	1.15e-230	1.4e-228
Archi 10	900.64	1450.74	6.4e-195	0.0

even in small architectures. To identify similarities in the behavior of the different metaheuristics and determine if there are statistically significant differences in their performances when changing the area size, the inner architecture, and the number of obstacles, we have conducted the nonparametric Kruskal-Wallis test, considering all the algorithms, with a significance level of 0.05. This was followed by the pairwise Tukey HSD test for pairwise comparisons. The findings outlined in Table 6.3 and Table 6.4 demonstrate that p-values below 0.05 signify a notable difference in performance between the pairs of algorithms under examination in the case of the Tukey HSD test, and among all algorithms in the case of the Kruskal-Wallis test.

2. **Examination of convergence graphs**
   Figures 6.1 and 6.2 display the convergence graphs of NSGA-II, SPEA-II, MOPSO, and MOGWO across ten indoor architectures over 450 iterations. These graphs exhibit the hypervolume and IGD indicators for the archives of SPEA-II, MOPSO, MOGWO, and the Pareto front of NSGA-II. Across all architectures, NSGA displayed consistent improvement from the initial iteration until convergence, achieving a relatively good hypervolume and slightly less good IGD for medium and large architectures compared to SPEA-II and MOGWO. SPEA-II achieved superior solutions compared to those reported in Table 6.2 since the number of iterations was used as the stopping criterion rather than the maximum fitness evaluation. While this elongated the execution time of SPEA-II, it enabled outperforming all other algorithms in terms of hypervolume and IGD across various instances. Despite these good outcomes, SPEA-II displayed disruptions in certain convergence curves due to the archive size limit.

Table 6.4: The Tukey pairwise comparison

		nsga-ii,spea-ii		nsga-ii,mogwo		nsga-ii,mopso		spea-ii,mogwo		spea-ii,mopso		mogwo,mopso	
		HV	IGD	HV	IGD	HV	IGD	HV	IGD	HV	IGD	HV	IGD
Archi 1	meandiff	-0.0192	0.1238	-0.0007	-0.0456	0.0403	-0.2145	-0.0199	0.0782	0.0212	-0.0907	-0.0411	0.1689
	p-value	-0.0	-0.0	0.356	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	True	True	False	True	True	True	True	True	True	True	True	True
Archi 2	meandiff	-0.0007	-0.1112	-0.0026	0.272	0.072	-0.0563	-0.0026	0.1608	0.0713	-0.1675	-0.0739	0.3283
	p-value	0.9829	-0.0	0.5513	-0.0	-0.0	-0.0	0.5513	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	False	True	False	True	True	True	False	True	True	True	True	True
Archi 3	meandiff	-0.0051	-0.0	0.0016	0.272	0.082	-0.2466	-0.0035	0.0091	0.0568	-0.2214	-0.0804	0.2557
	p-value	0.0	1.0	0.4695	-0.0	-0.0	-0.0	0.0059	0.0139	-0.0	-0.0	-0.0	-0.0
	reject	True	False	False	True	True	True	True	True	True	True	True	True
Archi 4	meandiff	-0.012	0.0594	0.0007	-0.0119	0.0688	-0.2808	-0.0113	0.0475	0.0769	-0.2466	-0.0682	0.2689
	p-value	-0.0	-0.0	0.8232	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	True	True	False	True	True	True	True	True	True	True	True	True
Archi 5	meandiff	-0.0038	-0.0547	-0.0042	0.084	0.0914	-0.1974	-0.0042	0.0293	0.0877	-0.2522	-0.0956	0.2815
	p-value	0.0291	-0.0	0.0122	-0.0	-0.0	-0.0	0.0122	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	True	True	True	True	True	True	True	True	True	True	True	True
Archi 6	meandiff	-0.0017	-0.0542	-0.0043	0.079	0.0731	-0.1571	-0.0059	0.0248	0.0714	-0.2113	-0.0774	0.2361
	p-value	0.3959	-0.0	0.0004	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	False	True	True	True	True	True	True	True	True	True	True	True
Archi 7	meandiff	-0.0005	-0.0535	-0.0081	0.0847	0.0922	-0.2046	-0.0086	0.0312	0.0917	-0.258	-0.1003	0.2893
	p-value	0.9738	-0.0	0.0	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	False	True	True	True	True	True	True	True	True	True	True	True
Archi 8	meandiff	-0.0045	-0.0482	-0.0045	0.0979	0.1023	-0.1806	-0.0091	0.0497	0.0977	-0.2288	-0.1068	0.2784
	p-value	0.0205	-0.0	0.0208	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	True	True	True	True	True	True	True	True	True	True	True	True
Archi 9	meandiff	-0.0053	0.019	-0.0009	0.008	0.0725	-0.2321	-0.0062	0.027	0.0672	-0.2132	-0.0734	0.2401
	p-value	0.0	0.0	0.8622	0.0209	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	True	True	False	True	True	True	True	True	True	True	True	True
Archi 10	meandiff	-0.0051	-0.1084	0.0125	0.2563	0.0863	-0.0775	0.0074	0.1479	0.0812	-0.1859	-0.0738	0.3338
	p-value	0.0623	-0.0	0.0	-0.0	-0.0	-0.0	0.0017	-0.0	-0.0	-0.0	-0.0	-0.0
	reject	False	True	True	True	True	True	True	True	True	True	True	True

This size limit was set to the population size and yet the deletion mechanism has affected the quality of the generated archive in different scenarios. Similar disruptions were observed in the hypervolume convergence curves of MOGWO for the larger architectures (archi 2 and archi 10). Nevertheless, based on the values presented in Table 6.2 and across various instances, MOGWO consistently produced high-quality solutions from the initial iterations, eventually achieving the best values for hypervolume and IGD. MOPSO faced premature convergence and achieved the least hypervolume and IGD values across all architectures, exhibiting a significant performance gap compared to the other algorithms.

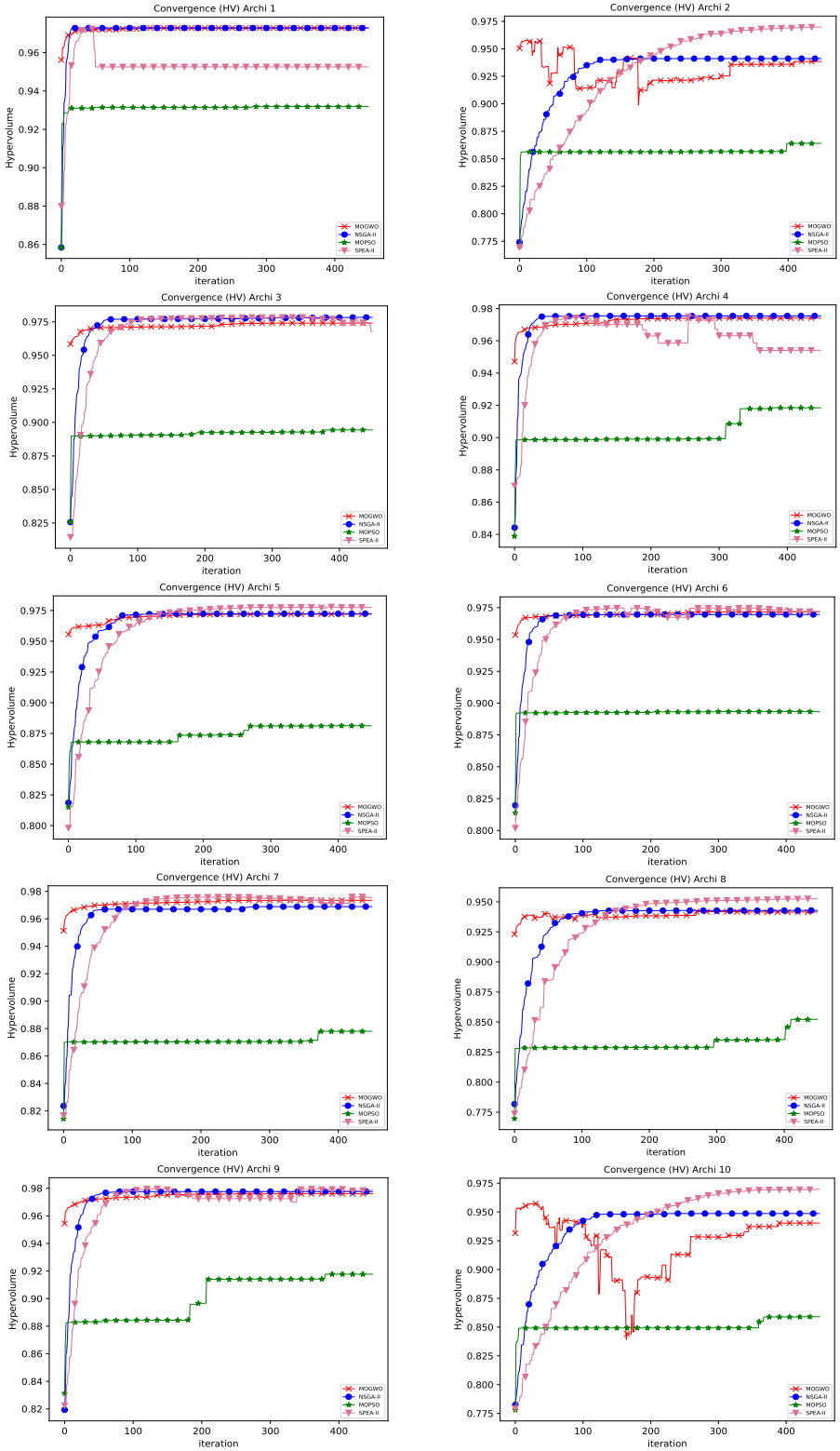


Figure 6.1: Convergence graphs of hypervolume indicator for the 10 indoor architectures

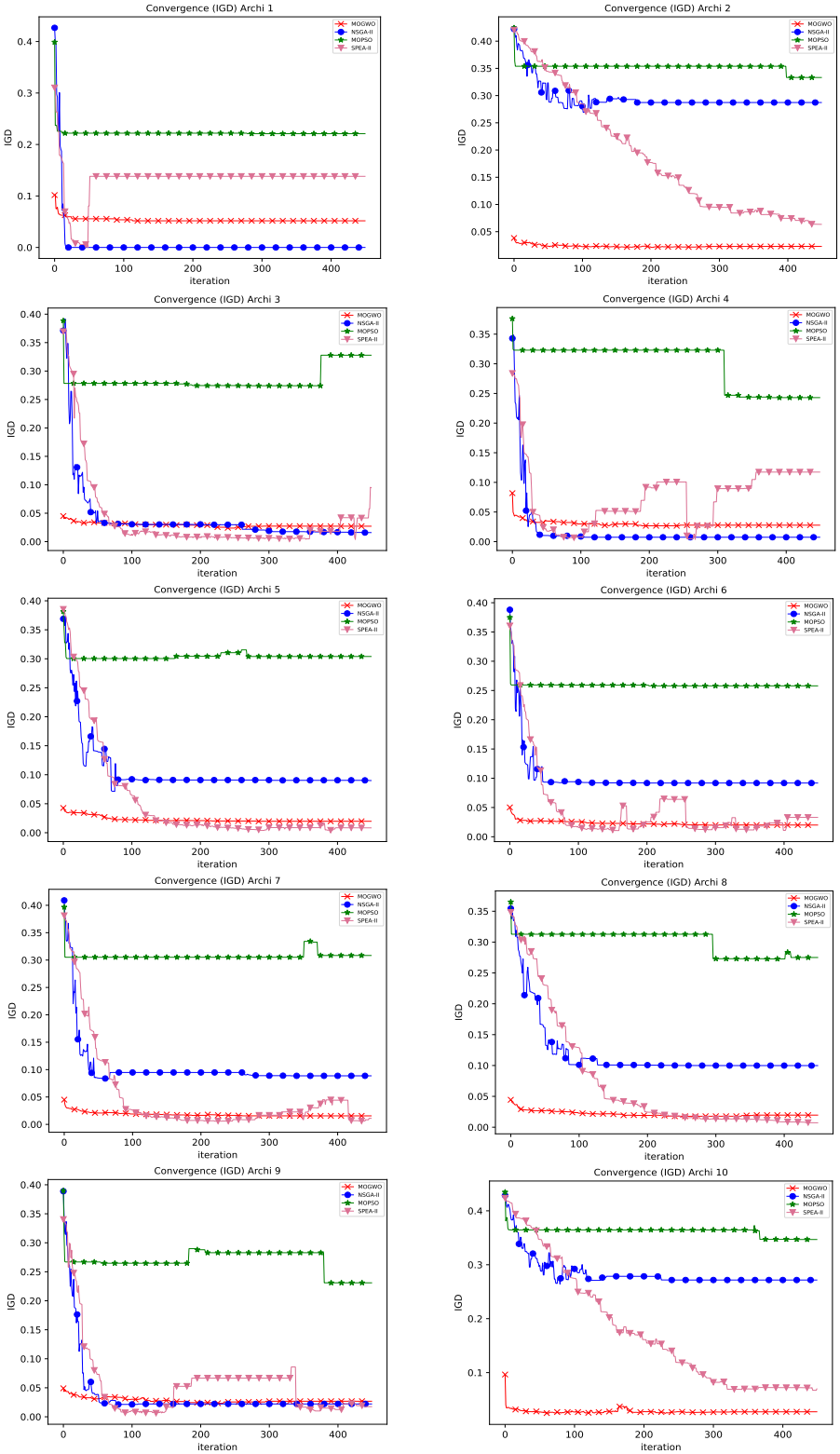


Figure 6.2: Convergence graphs of IGD indicator for the 10 indoor architectures

6.3 LHNSGA-II: hybrid NSGA-II with local search for WSN deployment

6.3.1 HNSGA-II: hybrid NSGA-II

Based on the results in Table 6.2, NSGA-II and MOGWO have achieved the best solutions across small, medium, and large architectures. Furthermore, the convergence graphs of hypervolume and IGD reveal that MOGWO consistently produces high-quality solutions from its initial iterations. Yet, the quality of solutions could decrease due to the restricted size of the archive and the deletion mechanism, which might eliminate superior topologies if the number of non-dominated solutions surpasses the archive capacity. Based on these findings, we have introduced a novel hybrid NSGA-II, called HNSGA-II, which combines NSGA-II with MOGWO. HNSGA-II employs a sequential hybridization approach, where MOGWO operates within the initial iterations to produce an archive of high-quality non-dominated solutions. Subsequently, this archive integrates into the initial population of NSGA-II, which continues execution for the remaining iterations. This integration aids NSGA-II in enhancing its exploitation capability while preserving population diversity.

6.3.2 LHNSGA-II: hybrid NSGA-II with local search

Another approach, LHNSGA-II, has been introduced by integrating a local search into HNSGA-II. This enhancement aims to further improve both the coverage and the cost efficiency of the resulting WSN topologies. Fig. 6.3 displays the flowchart depicting LHNSGA-II. The proposed local search involves two phases. The initial phase focuses on enhancing network coverage, while the subsequent phase aims to reduce deployment costs. In the first phase, a group of sensors is randomly selected and relocated to the first deployment zone from a subset of potential zones that enhance the overall coverage. This subset of potential zones is chosen based on the connectivity constraint. To determine this subset, a distinction is made between sensors that function as cut nodes within the WSN and the remaining sensors. Cut nodes are sensors that, if moved, could result in the WSN being split into multiple disjoint sub-networks. Therefore, in the case the selected sensor is a cut node in the WSN, each potential deployment zone is within the intersection of all the deployment zones reached from neighboring sensors of the selected sensor; otherwise, the candidate deployment zones result from the union of all deployment zones reached from neighboring sensors of the selected sensor. During the second phase, sensors that are

not cut vertices in the WSN communication graph and do not contribute to the network coverage are deactivated. Algorithm 16 illustrates the pseudo-code of the suggested local search.

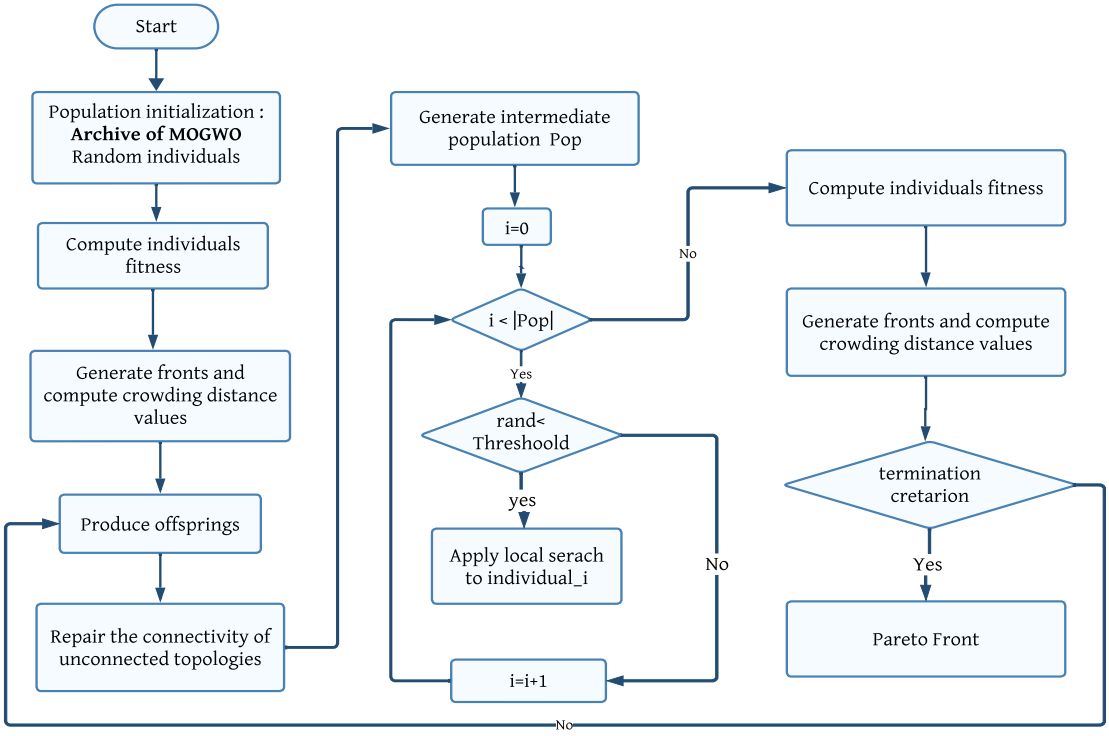


Figure 6.3: The global architecture of the proposed approach LHNSGA-II

6.3.3 Performance evaluation

In this experimental test, NSGA-II, MOGWO, HNSGA-II, and LHNSGA-II were first executed to generate WSN topologies within the largest (archi 2 and archi 10), medium-sized (archi 8), and smaller (archi 4) architectures. Figures 6.4 and 6.5 showcase box plots presenting the hypervolume and IGD, respectively, derived from 25 independent runs for each architecture. For clarity, the figures denote HNSGA-II as HNSGA and LHNSGA-II as LHNSGA. LHNSGA-II consistently produces the highest hypervolume and the lowest IGD values across all runs, with a condensed box indicating similar quality among all generated Pareto fronts. Furthermore, a marginal difference between LHNSGA-II and HNSGA-II highlights the significant performance enhancement achieved through the fusion of MOGWO and NSGA-II. This hybridization notably elevates the ability to generate high-quality solutions, even in larger indoor architectures. The algorithm ranking based on the reported results is LHNSGA-II $>$ HNSGA-II $>$ MOGWO $>$ NSGA-II.

Algorithm 16: The proposed local search

Data: *Topology* /* a WSN encoded in binary vector*/
$Nb_{sensors}$ /* Number of sensors to move*/
Targets /* the list of targets to cover*/
Result: *Topology*: /* Optimized *Topology* */
/* Part 1: optimize the network coverage */
graph ← Create deployment graph (*Topology*);
for *i in range($Nb_{sensors}$)* **do**
    cutVertices ← Articulation points of *graph*;
    candidates ← all sensors deployed in *Topology*;
    if |*candidates*| > 0 **then**
        $sensor_{move}$ ← Select a sensor randomly from *candidates*;
        neighbors ← *graph*.neighbors(*$sensor_{move}$*);
        cTargets ← subset of *Targets* that are covered only by *$sensor_{move}$*;
        candidateZones ← ∅;
        if *$sensor_{move}$* ∈ *cutVertices* **then**
            czones ← ∅;
            for *neighbor in neighbors* **do**
                czones.append(list(deployment zones connected to *neighbor*));
            end
            candidateZones ← intersection between all lists in *czones*;
        else
            for *neighbor in neighbors* **do**
                candidateZones ←append all deployment zones connected to *neighbor*;
            end
        end
        shuffle(*candidateZones*);
        for *zone in candidateZones* **do**
            zTargets ← subset of *Targets* that will be covered by *zone*;
            if |*zTargets*| > |*cTargets*| **then**
                Topology[*$sensor_{move}$*] ← 0;
                Topology[*zone*] ← 1;
                graph ← Create deployment graph (*Topology*);
                break;
            end
        end
    end
end

Algorithm 17: The continuation of the proposed local search

/* Part 2: reduce deployment cost*/

cutVertices ← Articulation points of *graph*;

for *j in range of* |*candidates*| **do**

  if *Topology*[*j*] == 1 *and j not in cutVertices and j is not dominated* **then**

    Topology[*j*] ← 0;

    graph ← remove sensor *j*;

    cutVertices ← Articulation points of *graph*;

  end

end

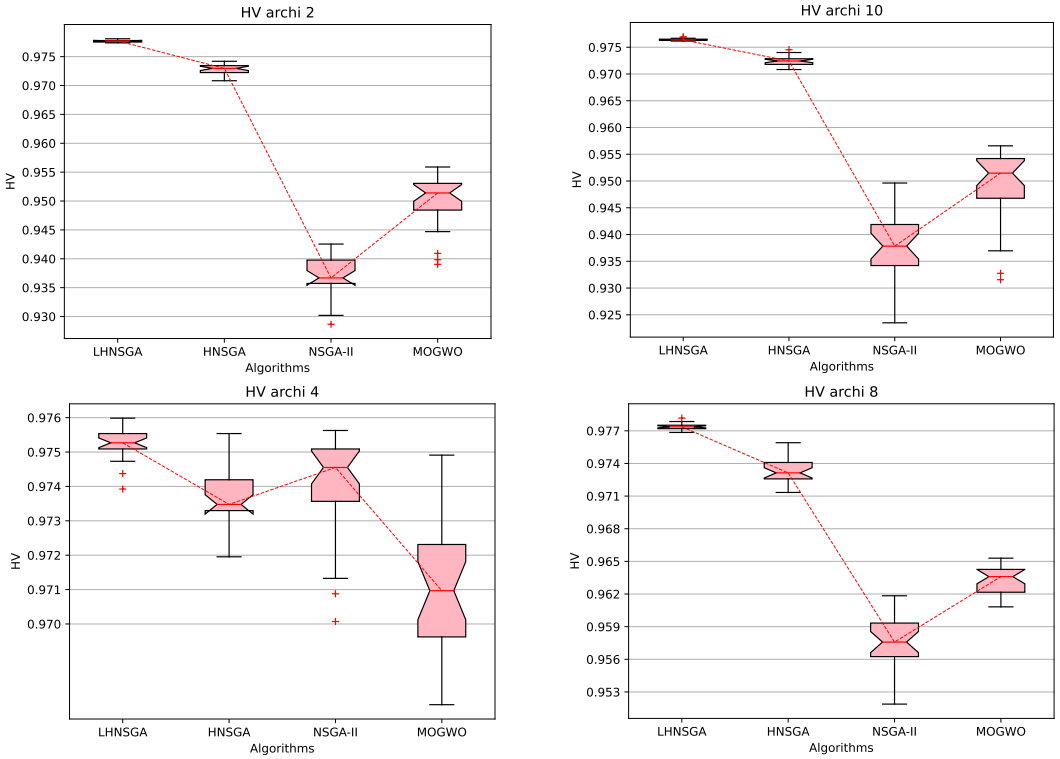


Figure 6.4: Box plot representations of hypervolume results for architectures 2, 4, 8, and 10.

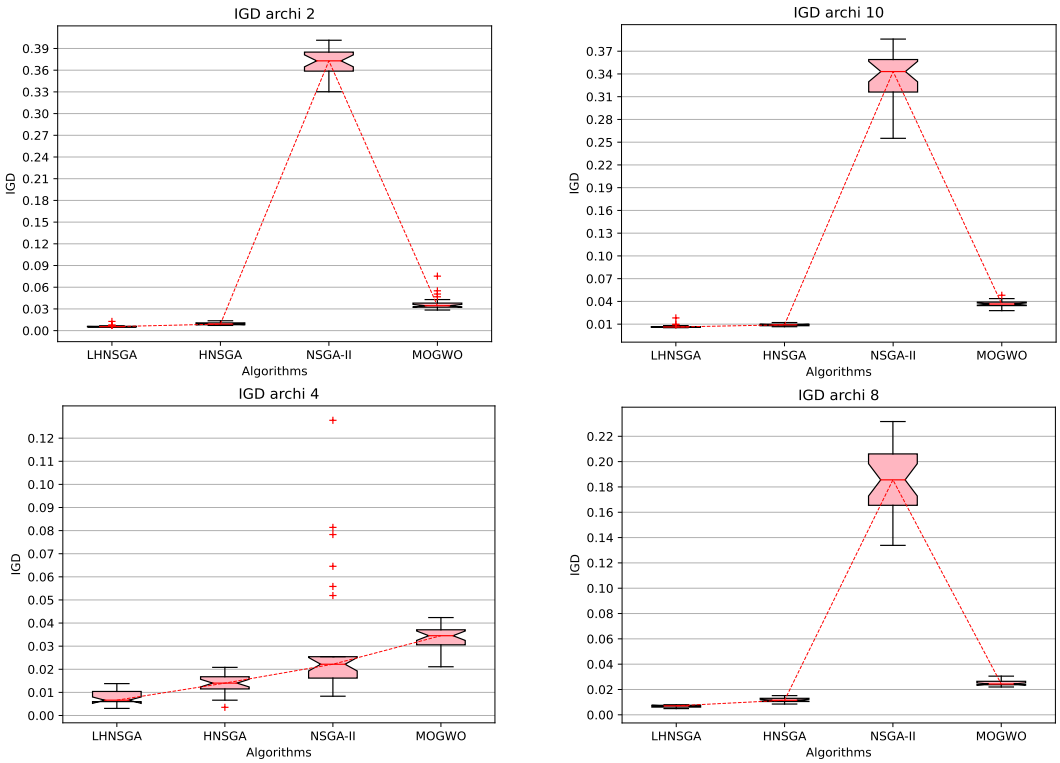


Figure 6.5: Box plot representations of IGD results for architectures 2, 4, 8, and 10.

After conducting tests on indoor architectures, we tested these methods on the BIM models presented in the previous chapter.

6.3.3.1 Analysis of the numerical results

Figures 6.6 and 6.7 illustrate boxplots of Hypervolume and IGD across 25 independent runs for all scenarios. In each scenario, LHNSGA-II obtained the superior values for both hypervolume and IGD. The resulting Pareto fronts align within a relatively narrow range which indicates that the results are stable and consistent throughout numerous runs and across all scenarios. HNSGA-II exhibits a minor performance difference compared to LHNSGA-II, yet their resulting Pareto fronts share several solutions as seen in IGD box plots. This observation validates the effectiveness of the proposed hybridization. The difference between LHNSGA-II and HNSGA-II is attributed to the inclusion of local search which improves further both coverage and deployment costs of certain topologies. Although the performance of MOGWO and NSGA-II is acceptable across scenarios, their hybridization significantly enhances the quality of the generated Pareto front.

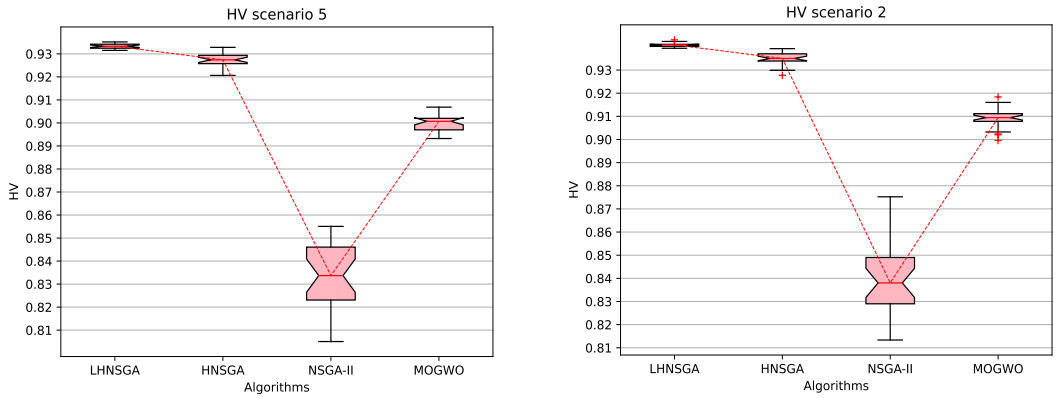
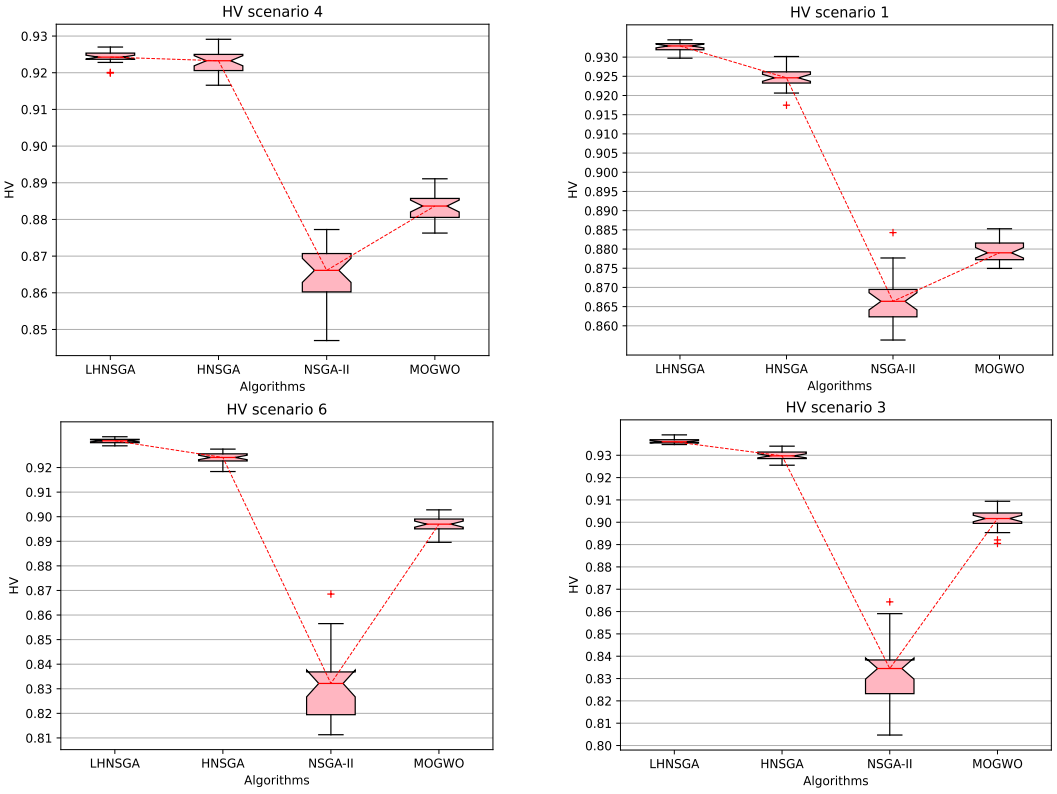


Figure 6.6: Box plot representations of hypervolume results for the 6 scenarios.



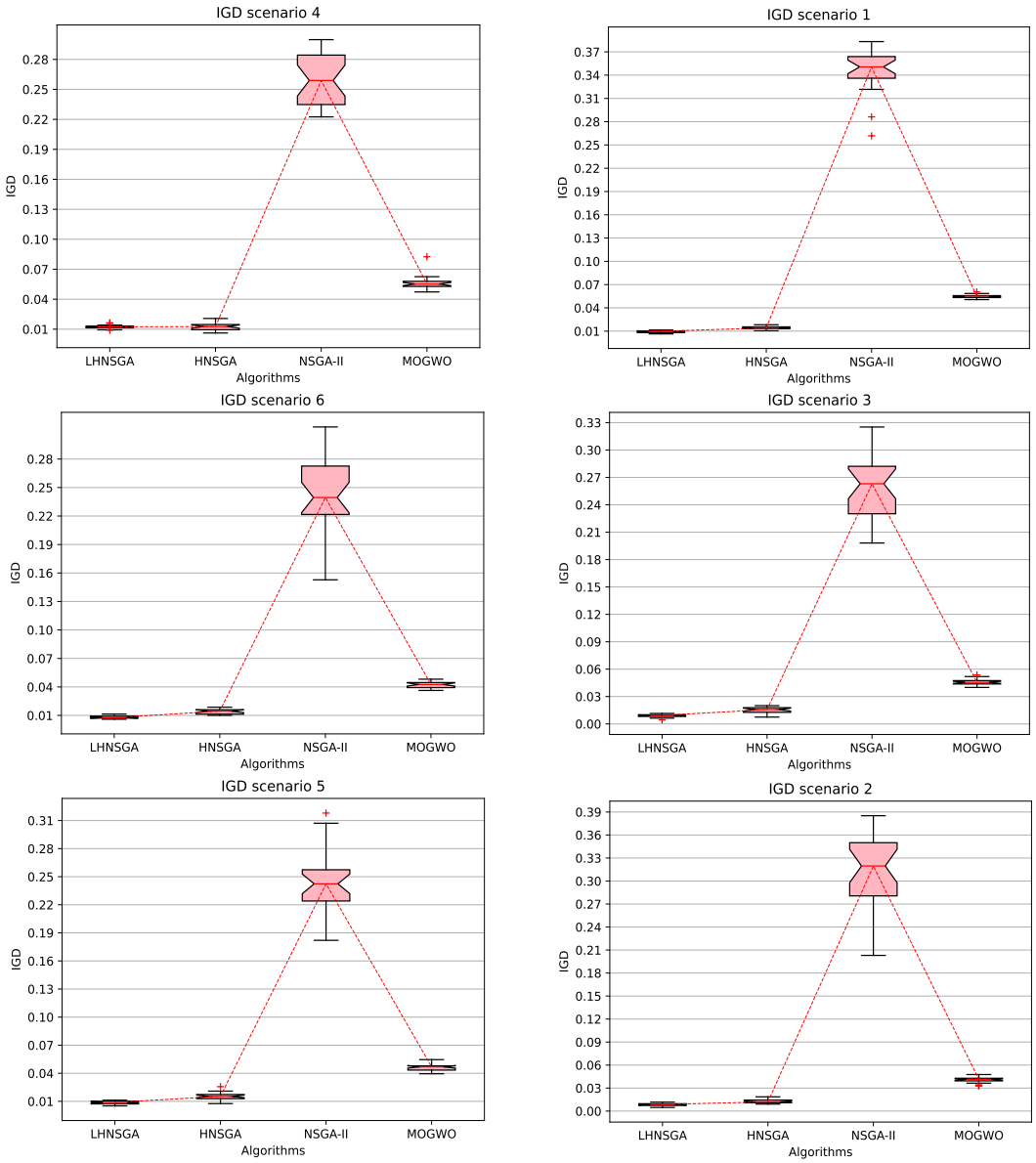


Figure 6.7: Box plot representations of IGD results for the 6 scenarios.

6.3.3.2 Examination of convergence graphs

Figures 6.8 and 6.9 illustrate the convergence graphs for hypervolume and IGD, respectively. These graphs reveal that initially, MOGWO, HNSGA-II, and LHSNGA-II demonstrate similar behaviors in the first quartile of iterations. However, as iterations progress, distinct behaviors emerge. MOGWO shows rapid convergence across all scenarios, while HNSGA-II displays slower improvement until it aligns with LHNSGA-II in the later iterations. The convergence graph of LHNSGA-II has a rapid significant improvement by the end of the first quartile iterations due to the integration of local search.

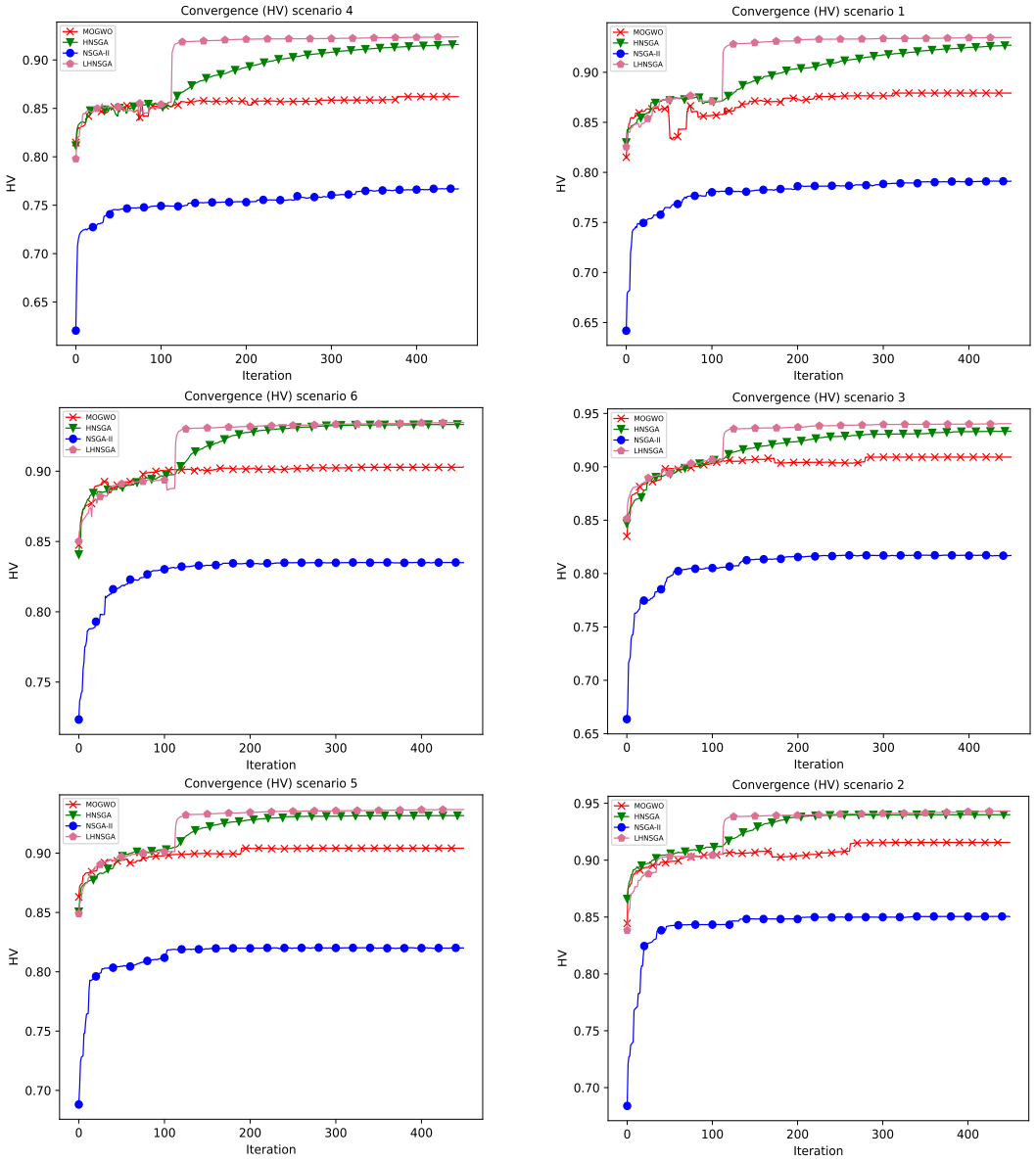


Figure 6.8: Convergence graphs of hypervolume indicator for the 6 scenarios

Consequently, our observation indicates that for evaluations with a limited budget, LHNSGA-II reaches high-quality solutions in fewer iterations. HNSGA-II showcases a similar performance to LHNSGA-II in later iterations, particularly in scenarios involving smaller target areas.

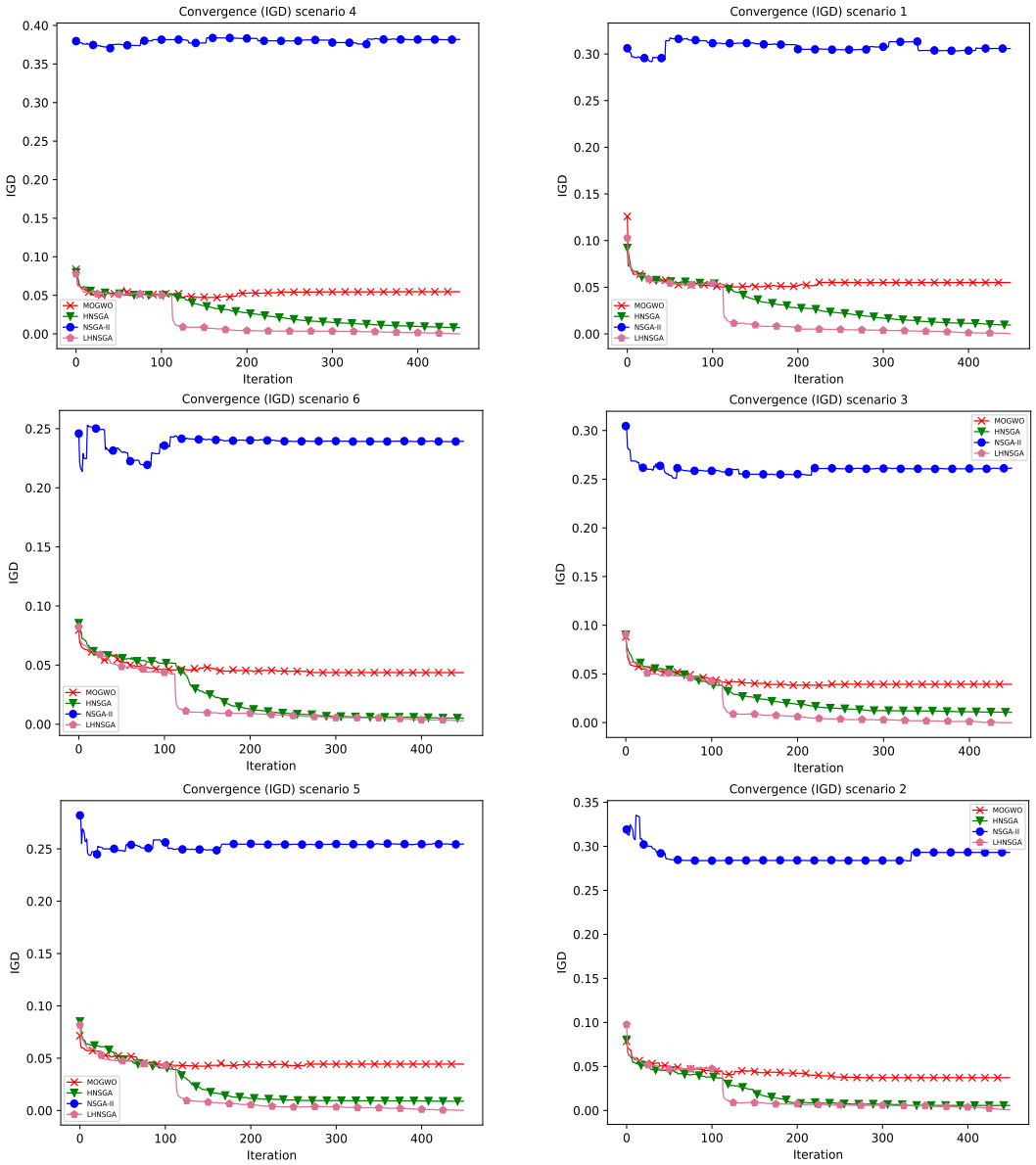
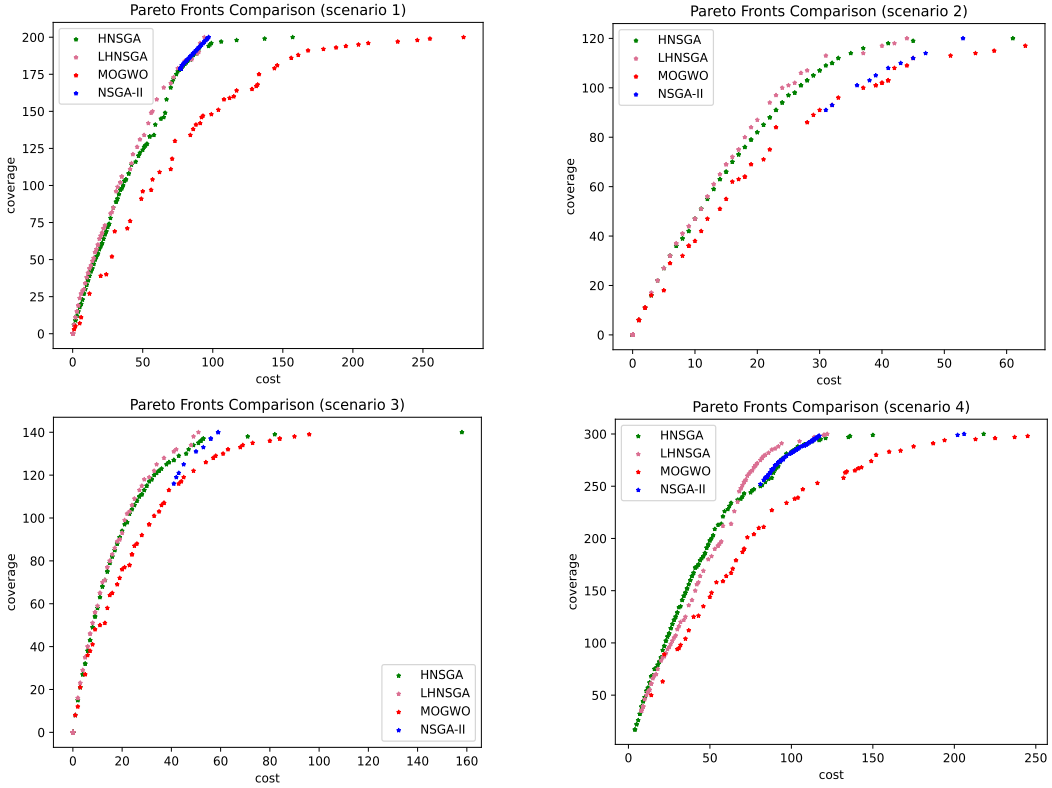


Figure 6.9: Convergence graphs of IGD indicator for the 6 scenarios

6.3.3.3 Examination of Pareto fronts

Fig. 6.10 illustrates the Pareto fronts of LHNSGA-II, HNSGA-II, NSGA-II, and MOGWO. Each Pareto front includes the best non-dominated WSN topologies for each scenario. On the X-axis lies the deployment cost of the generated topologies, while the Y-axis denotes the coverage, representing the number of target areas covered. In these figures, the positioning of the Pareto front serves as an initial indicator of its significance relative to other fronts. The Pareto front situated above the other fronts denotes that the associated method is proficient in generating topologies with equivalent coverage to those generated

by other methods, albeit at a lower deployment cost. Regarding the coverage objective, the Pareto fronts of LHNSGA-II, HNSGA-II, and MOGWO exhibit a balanced distribution, including WSN typologies that cover a wide range of coverage values (from smaller to larger values). Conversely, the Pareto fronts generated by NSGA-II encompass primarily solutions with high coverage and discard those with lower coverage. This highlights the limited ability of NSGA-II to explore the search space compared to other methods. In all scenarios, MOGWO demonstrates a consistent rate of increase between coverage and cost. Enhancements in coverage invariably result in higher deployment costs. This rate is significantly reduced in HNSGA-II, where the distance between topologies is notably reduced. This implies that HNSGA-II has the capacity to enhance coverage with significantly reduced deployment costs. However, in certain scenarios, HNSGA-II may display outlier solutions, characterized by a significant difference in deployment cost compared to the improvement in coverage. These outlier solutions are excluded from the Pareto front of LHNSGA-II through its second-stage local search, which entails removing sensors that do not significantly contribute to network coverage and connectivity. In almost all scenarios, all methods exhibit outlier typologies with high coverage and significant deployment costs, except for LHNSGA-II. In LHNSGA-II, the best topologies with high coverage values require nearly half the number of sensor nodes compared to other topologies with equivalent coverage in the Pareto fronts of other methods.



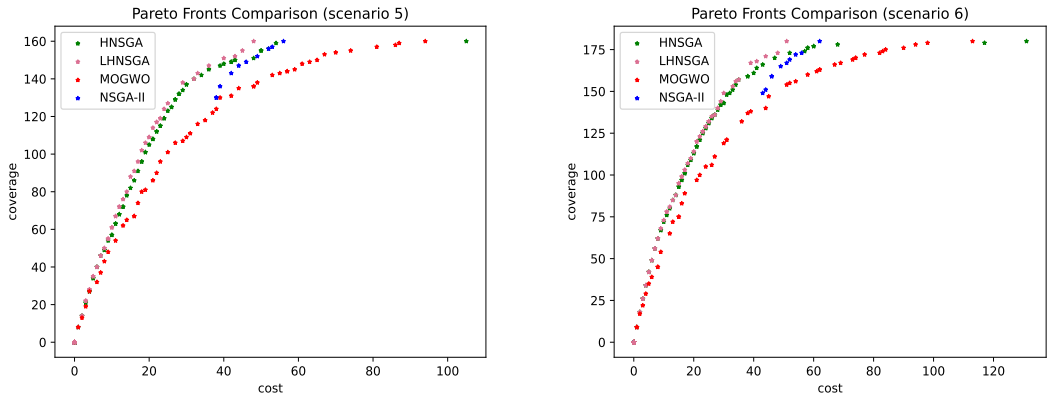


Figure 6.10: Pareto fronts of LHNSGA-II, HNSGA-II, NSGA-II, and MOGWO for the 6 scenarios

6.4 Conclusion

This chapter introduces a hybrid multi-objective algorithm designed to address the WSN deployment problem within smart buildings. The approach aims to optimize simultaneously both the network coverage and deployment cost under the connectivity constraint. To design the hybrid approach, we began by conducting a comparative study of widely used multi-objective algorithms referenced in the literature. This study focused on addressing the WSN deployment problem across ten indoor architectures that differ in area size, the number of obstacles, and the number of target points. The experimental results highlight the effectiveness of MOGWO and NSGA-II in resolving the problem within constrained evaluation budgets. In the subsequent phase of the research, a hybrid multi-objective approach named LHNSGA-II was developed to tackle the problem. This approach merges MOGWO with NSGA-II by incorporating the MOGWO archive into NSGA-II to boost its exploitation process. Additionally, a local search was designed to improve the generated topologies. This process involves relocating sensors to neighboring deployment zones while considering the connectivity constraint to increase the network coverage. Subsequently, non-dominated sensors that are not articulation nodes in the WSN communication graph were deactivated to minimize deployment costs. Six scenarios were generated using three BIM models to assess the performance of LHNSGA-II against HNSGA-II, NSGA-II, and MOGWO. The experimental results demonstrate the superiority of LHNSGA-II and highlight its efficacy in computing WSN topologies with optimized cost and coverage within smart buildings.

CHAPTER 7

DETECTION ZONE ESTIMATION FOR ELECTROMAGNETIC SENSORS IN SMART BUILDINGS

7.1 Introduction

Smart buildings integrate both contact and non-contact sensors into their sensing systems to enhance their functionalities. Contact sensors must be strategically placed in specific positions, as explained in Chapter 1. Therefore, this thesis focuses on deploying non-contact sensor nodes with a real detection zone. Among non-contact sensors, electromagnetic sensors are the most commonly used in smart buildings. They operate by emitting electromagnetic waves (such as infrared, microwave, or radio signals) into the deployment area and analyzing the reflected signals. These sensors include occupancy and motion sensors, which are essential components in almost all smart building applications. Accurately assessing the detection zone of these sensors is crucial as it directly impacts important building systems, including energy consumption, security, temperature, and lighting control systems.

In our deployment approaches, we have implemented the Elfes sensing model, widely acknowledged in the literature as the most acceptable model for approximating the detection zone of sensors compared to other simplistic models. However, this model does not consider the true impact of building obstacles on the detection signals emitted by electromagnetic sensors. This oversight could result in a misestimation of the actual coverage provided by the sensor, potentially leading to inaccuracies in the system's performance.

Apart from electromagnetic sensors, wireless communication systems depend on the transmission of diverse electromagnetic waves, encompassing radio waves, microwaves, and

infrared waves, which travel through the air to convey information. The key distinction among these waves lies in their frequency and wavelength, governing their propagation characteristics. Wave propagation refers to the movement and transmission of waves through the air, exposing them to several phenomena such as reflection, refraction, diffraction, absorption, and attenuation. Multiple propagation models have been devised to forecast the paths taken by electromagnetic signals, predict their attenuation, and determine key parameters like received signal strength, which denotes the signal's intensity after traversing from a transmitter to a receiver.

These propagation models are primarily employed for the design and optimization of networks and for evaluating their coverage. They can be either generic, such as empirical models, or site-specific, such as deterministic models. Empirical models are mathematical representations derived from observed measurements and real-world data collected in actual scenarios. Their main limitation lies in their effectiveness only for environments resembling where the measurements were taken. Deterministic models, encompassing techniques like ray tracing and finite-difference time-domain methods, are robust and dependable. They are grounded in the fundamental laws of electromagnetic wave propagation as described by Maxwell's equations [179].

While these models offer accuracy and the ability to calculate genuine characteristics of electromagnetic waves in a given propagation environment, their primary drawbacks include the necessity for high expertise and computational resources, as well as sensitivity to changes in the environment. To address this issue, various approaches have leveraged the potential of machine learning techniques for predicting electromagnetic signal propagation. Machine learning-based methods are considered suitable candidates to tackle the challenge of generability and are known for their computational efficiency.

This chapter explores the application of machine learning-based methods in modeling signal propagation and introduces a novel sensing model for electromagnetic sensors in smart buildings. The proposed sensing model can be integrated into deployment approaches to enhance their accuracy further. The objective is to evaluate the strength of detection signals in the deployment area and identify both covered and uncovered zones, while accounting for the heterogeneity of obstacles. The sensing model is developed using an enhanced encoder-decoder architecture, suitable for the considered problem, which involves a pixel-wise regression task.

7.2 Artificial neural networks for signal propagation

7.2.1 Multilayer Perceptrons

Multilayer perceptrons (MLPs) [180] are a subclass of artificial neural networks (ANNs) [181] composed of three layers: the input layer, hidden layers, and output layer. The input layer represents the initial features fed into the model. Hidden layers comprise all layers between the input and output layers and they allow estimating outputs from input data. The number of layers and neurons in the hidden layers constitutes model-specific hyperparameters. Each layer represents the input for the neurons in the subsequent layer. Within each neuron, these inputs are multiplied by weights, summed with a bias, and then subjected to an activation function to introduce non-linearity to the network as depicted in Fig. 7.1. Common activation functions include ReLU, sigmoid, Tanh, and leaky ReLU. The weights of each node are updated through the backpropagation method. The training process involves computing the optimal weights to minimize the error between the real output and the estimated output for unseen data.

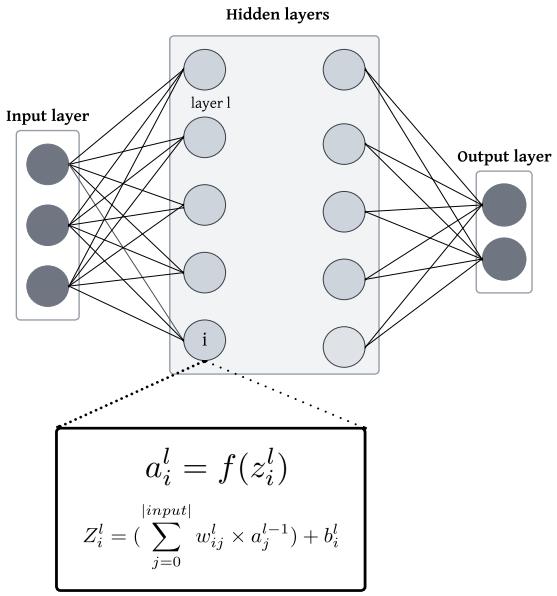


Figure 7.1: Standard architecture of Multilayer perceptrons

MLPs have been used in the literature to tackle various problems related to wireless networks namely in assessing some parameters related to radio propagation modeling [182] including Path loss (PL) [183, 184, 185, 186, 187], and received signal strength (RSS) [188, 189]. The main idea of these approaches is that the input layer incorporates various data

points associated with the environment. This includes information about the propagation environment, such as its type, as well as other details related to signal propagation, such as transmitter frequency, transmitter height, receiver positions, the distance between transmitter and receivers, the count of reflections and diffractions, and signal-related phenomena along with the corresponding measured values (RSS or PL). Models could consider more complex inputs related to signals and environments, such as the orientation of walls and their impact on signal reflection in the case of signal propagation within indoor environments. Generally, these models are trained using simulated data, as they are data-driven approaches that require a substantial amount of data for effective training. Conducting real-world measurements can pose a significant challenge due to the need for extensive data. The objective of the model is to accurately predict values of RSS, PL, or other signal-related measurements for unseen data.

Limitations: While MLPs have demonstrated success in certain scenarios, they come with several drawbacks and limitations. One significant constraint when addressing signal propagation-related problems is the lack of consideration for spatial information and local patterns characterizing the area surrounding the transmitter and the relationships between nearby zones in the propagation environment. Additionally, MLPs do not incorporate parameter-sharing which is an efficient method for learning mappings between spatial features and the output.

7.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) [190] are another type of ANNs designed primarily for processing grid-structured data, such as images and videos. These models find extensive application in computer vision tasks, notably in image segmentation and object detection. A standard CNN comprises five key components: the input layer, convolutional layers, pooling layers, fully connected layer, and the output layer. The input data to a CNN is represented as tensors, defined by their width, height, and number of channels. Convolutional layers aim at identifying patterns and extracting features by convolving the input tensor with filters to generate a feature map. Here, a filter is a 2D stacked kernel, with the number of kernels matching the number of channels in the tensor, and the number of output channels equal to the number of filters. Pooling layers are employed to reduce the complexity of the data by downsampling its spatial dimensions and retaining only essential features. Initial convolutional layers focus on extracting basic patterns, while subsequent layers extract more complex and meaningful patterns. In the final step, the complex features derived from the preceding convolutional layers are flattened and

connected to a fully connected layer to predict the class of the input grid. Fig 7.2 depicts the architecture of a CNN.

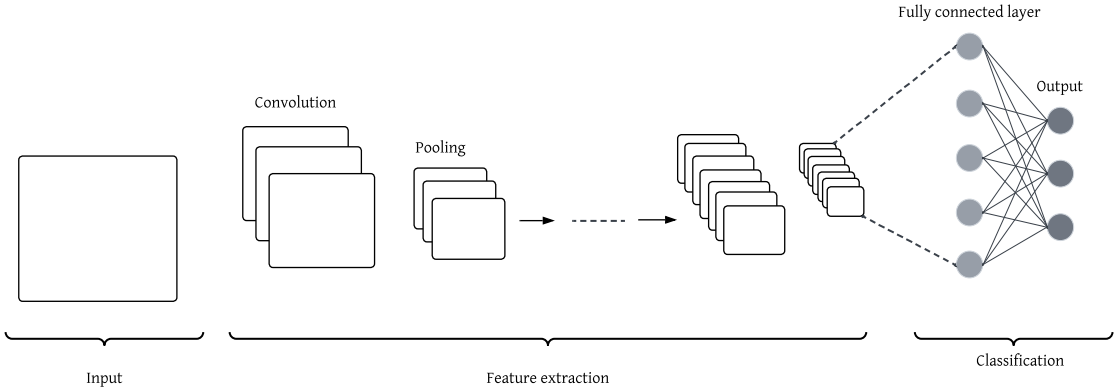


Figure 7.2: Standard architecture of a convolutional neural network

Similar to MLPs, CNNs have been employed in literature to address challenges associated with signal propagation. These challenges involve estimating signal characteristics such as PL [191, 192], and RSS [193, 194, 195]. Notably, the distinction among these approaches lies in the type of input features. CNN-based signal propagation modeling methods utilize grid-structured input to depict the propagation environment and the signal generators (antennas). To characterize the environment, various channels can be considered to represent features such as the arrangement of obstacles, elevations of placements, variations in obstacle types, locations of open spaces, and attributes of the signal generator, including the distance from other receivers and its frequency. The model output involves estimating signal characteristics in various zones.

Limitations: CNNs are effective at incorporating spatial information relevant to the propagation environment. Nonetheless, they encounter limitations in preserving these spatial correlations during output computation since they conclude with MLPs. In studies aiming to generate maps, such as the RSS map discussed in the work by Seretis et al. [194], authors applied a CNN model to obtain estimated RSS values for all receivers, next, they applied interpolation to generate the final RSS map. This approach could produce good results, yet it necessitates significant computational resources due to the huge number of parameters involved in the training process. Additionally, the integration of MLPs in CNNs results in a loss of information concerning cross-channel correlations and, consequently, some spatial details.

7.2.3 Convolutional Encoder Decoder

To address the mentioned limitations of CNNs, a variant of ANNs called Fully Convolutional Networks (FCNs) [196] has been developed. FCN is an adapted version of CNNs, where the traditional fully connected layer is replaced with a sequence of upsampling layers for pixel-wise predictions. In this scenario, the output is not a class label for the entire image, as in CNNs, but rather another grid where each pixel corresponds to a class label for the respective pixel in the input image. Convolutional Encoder-Decoder models represent a specific type of FCNs, involving a grid-structured input, an encoder, a decoder, and a grid-structured output. The encoder typically consists of a series of convolutional and pooling layers that extract and encode hierarchically features from the input data. The decoder is responsible for reconstructing the output based on the encoded features by iteratively increasing the dimension. It incorporates upsampling layers, skip connections, and convolutional layers. Upsampling layers enlarge the spatial dimension of the corresponding feature map, while skip connections concatenate the output of a specific encoder layer with its counterpart in the decoder to maintain the same dimension. This connectivity preserves the information that could be lost during the encoding phase. Fig. 7.3 illustrates the overall architecture of a convolutional encoder-decoder. U-Net [197] and SegNet [198] are examples of well-known Convolutional Encoder-Decoder architectures that have been proposed to solve computer vision problems.

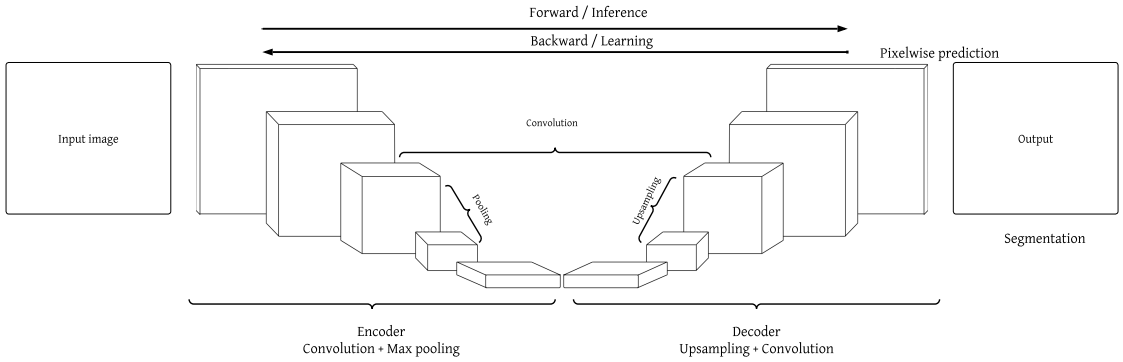


Figure 7.3: Standard architecture of a convolutional encode-decoder

Convolutional encoders-decoders are primarily employed in the literature for semantic segmentation where the goal is to classify every pixel in a grid-structured input. This application proves advantageous in creating signal measurement maps, such as RSS maps, where each pixel in the output grid corresponds to the RSS in a specific zone of the propagation environment. Consequently, convolutional encoders-decoders enable the estima-

tion of RSS for a set of receivers within a single training process while considering the spatial correlation between channels. These models can be applied to estimate various signal measurements, as shown in [199] and [200]. In [199], the authors introduced FadeNet, a U-Net-like model designed to predict large-scale fading for 5G millimeter-wave cellular networks. The inputs of FadeNet describe outdoor environments by incorporating various factors such as building height, foliage, and terrain, collected from three distinct cities. The model's output corresponds to the large-scale fading associated with the given input. For training data, the output, represented by large-scale fading maps, was generated using a Ray tracer solver. An additional application of convolutional encoders-decoders for evaluating RSS in indoor environments was introduced in [201]. In this work, the authors employed a U-Net like architecture, named SDU-Net, initially designed for semantic segmentation in medical images. SDU-Net has demonstrated exceptional performance in addressing signal propagation-related problems, as elaborated in the subsequent section.

7.3 Proposed sensing model in smart buildings

The sensing process is a pixel-wise regression task which involves finding the strength of the reflected detection signal from every emplacement within the considered area. Building upon this principle, our proposed sensing model is based on the SDU-Net model [202] (U-Net Using Stacked Dilated Convolutions), a variant of the U-net model [197] adapted to pixel-wise regression. The objective of our sensing model is to identify covered and uncovered zones within the deployment area. This is achieved by generating a signal map that illustrates the strength of reflected detection signals from all positions within the deployment area. Accordingly, zones where the signal detection exceeds the sensor sensitivity threshold are deemed covered, otherwise the signal is interpreted as noise indicating that the corresponding zones are uncovered.

7.3.1 Input features

The operational principle of electromagnetic sensors involves emitting electromagnetic waves into the deployment area. When a person enters the sensor's detection zone, they may reflect some of these waves back towards the sensor. The sensor then analyzes various characteristics of the reflected signals, including frequency, amplitude, phase shift, and time delay, to extract information such as the distance to the person, their speed, and direction of movement. However, the effective detection range of electromagnetic sensors

in real-world scenarios can be significantly smaller than their theoretical maximum range of 10 meters due to environmental factors and obstacles. To effectively assess and analyze the reflected signals, their strength must exceed the sensitivity threshold of the sensor; otherwise, the sensor interprets them as noise. Indeed, the strength of reflected signals is significantly influenced by the obstacles in the deployment area, which is likely to reduce the detection zone for the electromagnetic sensor. The impact of obstacles on the signal varies based on their construction materials (wood, glass, plasterboard, concrete, and brick), thickness, shape, and location. Therefore, it is important to consider all these attributes when evaluating the strength of reflected detection.

In this study, the input tensor describes the propagation area where the sensor is deployed. The tensor comprises four channels: permittivity, conductivity, distance, and line of sight (LoS) channels. The theoretical sensing ability of sensors cannot exceed 10 m, so we assume that the maximum detection zone of a sensor is a square of 20×20 m^2 where the sensor is at its center, and the distance between the sensor and each edge of this square is 10 m. To model this area, we assume that each channel in the tensor is a matrix of 400×400 cells, so each cell in a channel represents 5×5 cm^2 in the propagation area. This level of granularity allows us to effectively model obstacle thickness, as all building obstacle thicknesses can be approximated to multiples of 5 cm.

The permittivity channel serves as a visual representation of the deployment area, providing a detailed 2D plan where each cell corresponds to a specific location within the deployment area. Each obstacle is delineated within the channel by a rectangle, with the length of the rectangle indicating the length of the obstacle. Here, each 5 cm of the obstacle is represented by one single cell in the permittivity channel. Moreover, the width of the rectangle corresponds to the thickness of the obstacle. All cells encompassed within this rectangle are then assigned the permittivity value corresponding to the construction material of the obstacle they represent. The process of filling the conductivity channel follows the same method as the permittivity channel, except that it assigns conductivity values based on the materials involved. When combined, the permittivity and conductivity channels offer a comprehensive representation of the deployment area's layout and identify the construction material type for each obstacle. Consequently, our sensing model will have the capability to establish connections between the spatial arrangement, obstacles materials and thicknesses, and the strength of detection signals. The distance channel provides information on the distance between the sensor and every position within the deployment area, allowing the model to assess the impact of distance on signal strength. Meanwhile, the Line of Sight (LoS) channel assists the model in identifying positions within the deployment area that are either in direct line of sight from the sensor or obstructed by one

or more obstacles. This feature enhances the model's capability to generalize to unseen environments [194] and achieve superior performance, particularly in complex scenarios. Fig. 7.4 illustrates the channels corresponding to the specified indoor area.

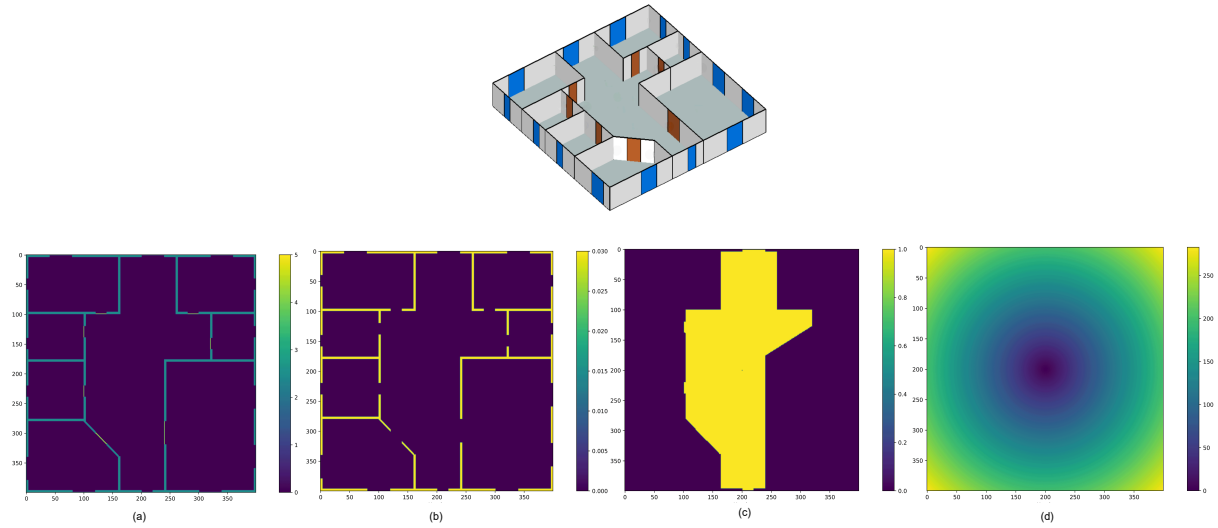


Figure 7.4: Tensor channels for the specified indoor area include: (a) permittivity channel, (b) conductivity channel, (c) Line of Sight (LoS) channel, and (d) distance channel.

7.3.2 Model output

The output of our sensing model represents a signal strength map, modeled by a tensor (matrix) of the same size as the input tensor (400×400). Each cell in the output tensor represents a position in the indoor environment and contains the strength value of the detection signal reflected from that corresponding position. Hence, any location with a signal strength value below the sensor's sensitivity threshold is considered uncovered, whereas those with a signal strength exceeding the sensitivity threshold are considered within its detection range. The process of generating this signal strength map is explained in detail in Section 7.4.1. Fig. 7.5 illustrates an indoor environment and its corresponding signal strength map.

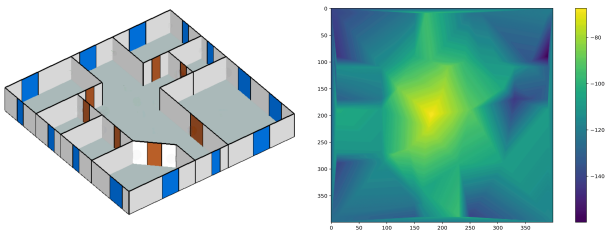


Figure 7.5: Example of an indoor environment and its corresponding signal strength map

7.3.3 SDU-Net based sensing model

The SDU-Net model is a variant of the well-known U-Net architecture. Its efficacy in addressing pixel-wise regression tasks is attributed to its ability to capture intricate patterns in the input data, thereby enhancing the precision and effectiveness of the sensing model. A standard U-Net architecture comprises five downsampling and upsampling layers. A downsampling layer consists of two sequential 3x3 convolutions, followed by the ReLU (rectified linear unit), and a 2x2 max pooling with a stride of 2 for downsampling. The output from the current downsampling layer serves as the input for the subsequent layer. Conversely, an upsampling layer utilizes a 2x2 up-convolution to reduce the number of channels by half and augment the dimension of the feature map. Subsequently, it concatenates the resulting feature map with its corresponding feature map in the encoder. The U-Net architecture has been successfully applied to address semantic segmentation [203] and pixel-wise regression tasks [204]. Nevertheless, it exhibits a limited receptive field, potentially leading to a decline in the correlation between pixels [202]. The receptive field represents the region in the input data that influences the activation of a specific feature in the output feature map, as illustrated in Fig. 7.6.

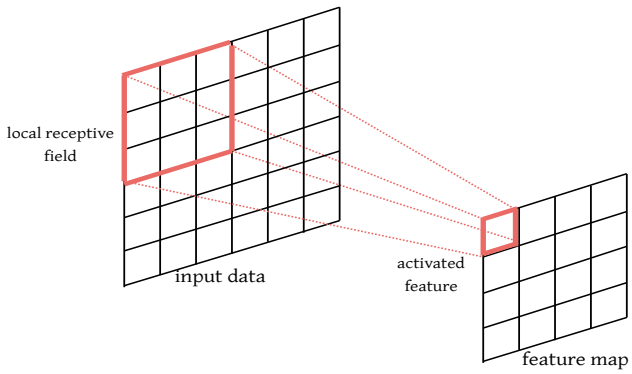


Figure 7.6: Receptive field influencing feature activation in the output map.

Our sensing model is based on the SDU-Net model. This model shares the same architecture as U-Net but introduces modifications to the encoding and decoding operations. In SDU-Net, the encoding operation diverges from U-Net as the second 3x3 convolution is replaced by an SDU-Net block. The SDU-Net block comprises five consecutive dilated convolutions (also known as atrous convolutions), each with specified output channels and dilation rates, as illustrated in Fig. 7.7. The output of all atrous convolutions is concatenated and either downsampled with max pooling in the case of a downsampling layer or up-convolved in the case of an upsampling layer before it will be transmitted to the

next-layer.

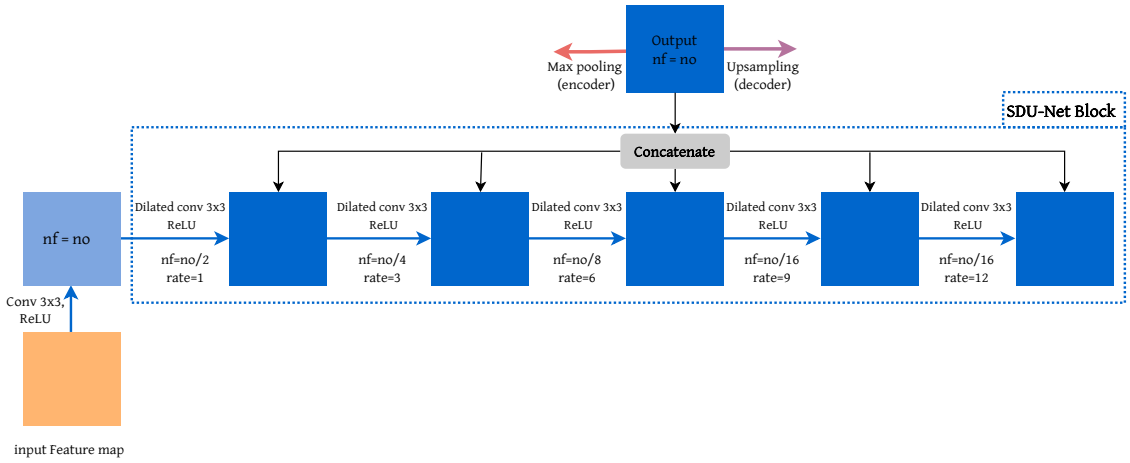


Figure 7.7: SDU-Net block with *no* representing the number of filters for the next layer, nf denoting the number of filters for the current convolution, and *rate* indicating the dilation rate.

Dilated convolutions represent a generalized form of standard convolutions, introducing dilation or "holes" between consecutive kernel elements. This dilation allows for the expansion of kernels without a concurrent increase in the number of parameters [202]. The specific number of spaces between kernel elements is defined by the dilation rate. Fig. 7.8 illustrates the application of dilated convolutions on an indoor environment, showcasing dilation rates of 1, 2, and 3. The integration of dilated convolutions into our sensing model yields manifold benefits, including:

- As dilated convolutions employ varying dilation rates, the model gains the capability to capture intricate patterns across diverse spatial scales within the deployment area, encompassing features like curved and sharp obstacles. This enables the sensing model to associate these patterns with signal phenomena, such as reflection and refraction, and estimate the strength of the resulting signal.
- Dilated convolutions enhance the recognition of correlations between distant points representing varied locations in the deployment area with different construction materials. This capability enables the sensing model to comprehend the behavior of the detection signal when encountering diverse types of construction materials and associate it with the resulting attenuation.
- Dilated convolutions possess a broad receptive field, enabling the model to identify the presence of obstacles with diverse sizes and shapes. This facilitates a comprehensive

understanding of how these obstacles influence the propagation behavior of detection signals.

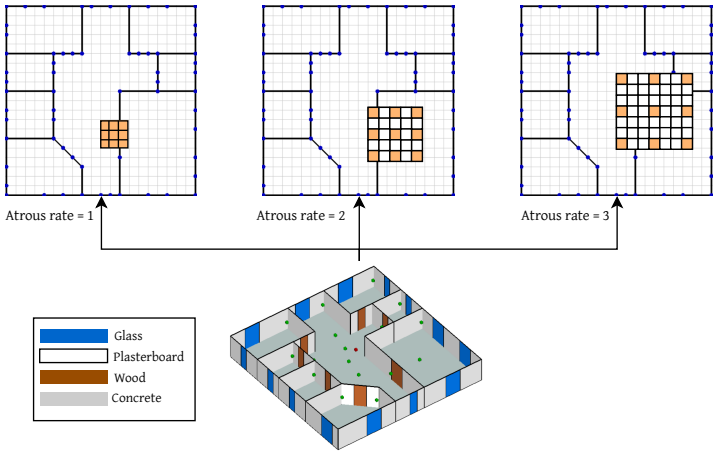


Figure 7.8: Application of dilated convolutions on an indoor environment, with dilation rates of 1, 2, and 3.

Fig. 7.9 illustrates the comprehensive architecture of the SDU-Net model utilized to convert the physics-based tensor input representing an indoor environment into a map depicting detection signal strength. The model receives a four-channel tensor with dimensions 400 x 400 cells as input, representing the deployment area of the sensor. Subsequently, five downsampling operations are performed to generate a feature map with dimensions 25 x 25 and 512 channels. This feature map integrates details about multiscale correlations across diverse resolutions. Subsequently, five upsampling operations are conducted, involving a 2x2 up-convolution followed by a standard 3x3 convolution and an SDU-Net block. The incorporation of skip connections at each upsampling layer enables the model to correlate indoor environmental traits (such as construction material, obstacle thickness, and shape) with the propagation behavior of the detection signal. After the last upsampling layer, a 1x1 convolution is applied to produce signal map.

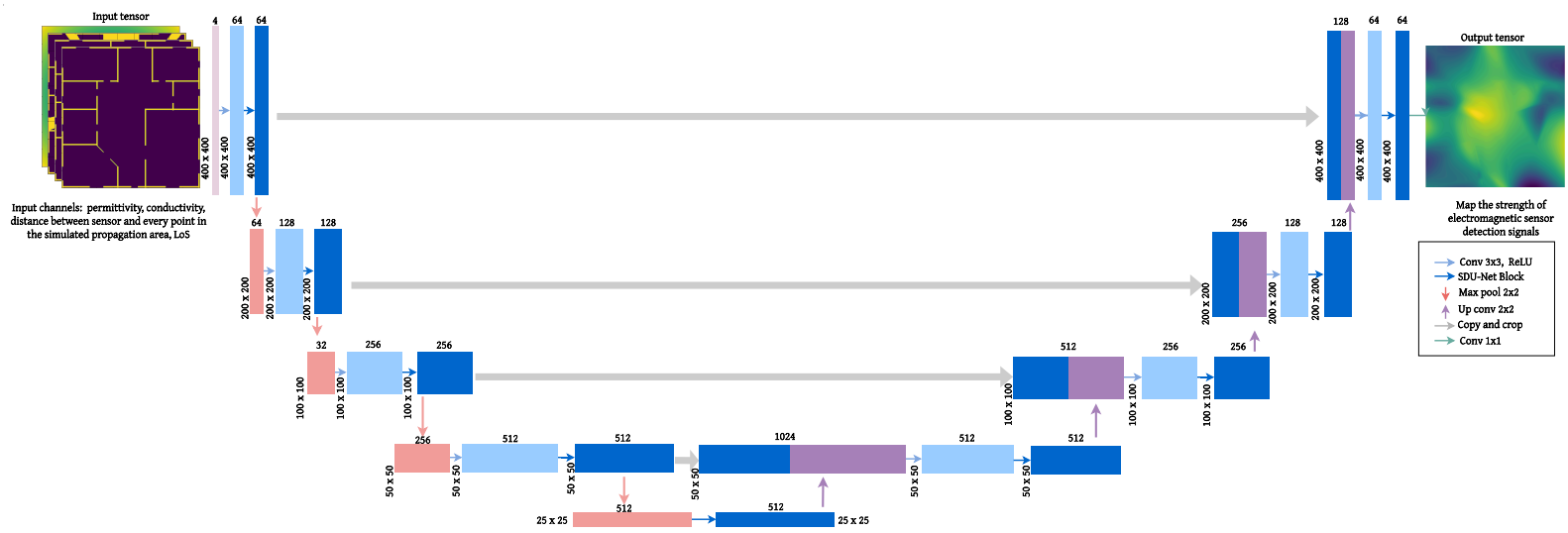


Figure 7.9: SDU-Net architecture used to estimate detection signal strength maps for electromagnetic sensors.

7.4 Performance evaluation

7.4.1 Dataset generation

1. **Simulating electromagnetic sensors:** Commercial 3D ray tracing solvers currently available are unable to simulate an electromagnetic sensor adequately in a single-phase simulation. These solvers typically conduct simulations involving the propagation of signals from a transmitter antenna to a set of receiver antennas. Once the propagation simulation is complete, the solver calculates essential characteristics of the received signals, such as signal strength, path loss, and time of arrival. However, when dealing with electromagnetic sensors, our focus is on the strength of reflected detection signals from each zone within the propagation area. This information is crucial for identifying covered zones (where the sensor can detect a human presence) and those that are not covered. An emplacement is considered covered by the sensor if the strength of the reflected detection signal from it surpasses the sensor's sensitivity threshold; otherwise, the emplacement is not covered. To accurately simulate the entire working principle of an electromagnetic sensor, it is essential to follow a two-step process. Firstly, simulate the propagation of detection signals in all directions within the indoor deployment area. Then, simulate the reflection of these signals back to the sensor. All simulations are conducted using Wireless InSite software, and the process unfolds as follows:

– **First stage: Simulation of the propagation of detection signals:** To model the propagation of detection signals, a transmitter (Tx) antenna is centrally positioned within each indoor scenario. The configuration of this Tx antenna aligns with the established characteristics of electromagnetic sensors in the market, as depicted in Fig. 7.10. Utilizing the ray tracing method, all signal paths are calculated. Subsequently, to evaluate signal strength across various locations, a network of receiver Rx antennas is distributed throughout the entire deployment area. Adding more receivers improves accuracy, yet it slows down processing. Therefore, each indoor scenario has a different number of distributed receivers based on its inner architecture.

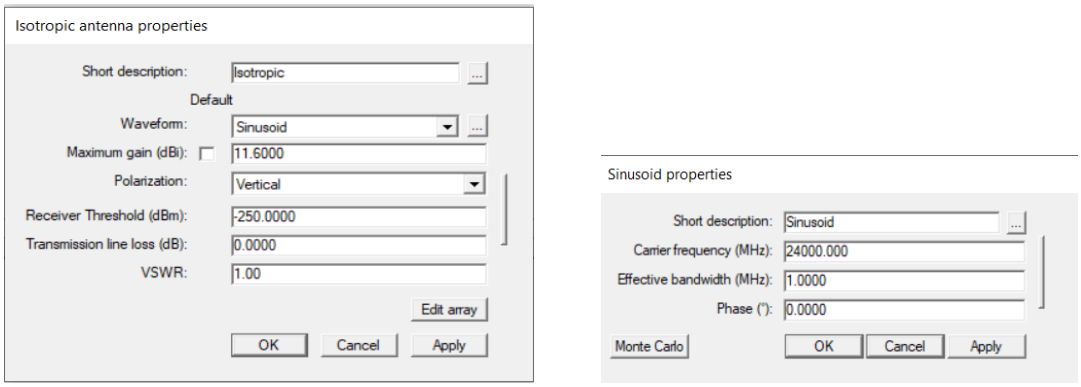


Figure 7.10: Characteristics of the Tx antenna employed in simulating an electromagnetic sensor.

– **Second stage: Simulation of the reflection of detection signals:** In the second stage, each RX antenna is transformed into a TX antenna, and the TX antenna simulating the sensor becomes an RX antenna (transceivers). Each TX antenna emits the signal received by the corresponding RX antenna to the sensor (the sensor is represented by the unique RX antenna at the center of the deployment area). Upon simulation completion, the RX antenna outputs the signal strength values received by each TX antenna, representing the signal strength of the reflected detection signals. In Fig. 7.11, an example of a complex indoor environment with multiple construction materials (concrete, wood, plasterboard, and glass) is illustrated. The red point signifies the TX antenna initiating the simulation by emitting electromagnetic signals in all directions, and the green points denote the set of RX antennas capturing the signal. For the second stage simulation, the RX antennas become TX antennas and transmit the received detection signal back to the unique RX antenna (sensor). The values of received signal strength and the positions of green points are employed to compute the received signal strength from all locations in the deployment area. This process utilizes linear interpolation, as shown in Eq. 7.1, to generate a map as depicted in Fig. 7.11 (right side).

$$y = y_1 + \frac{(x - x_1) \cdot (y_2 - y_1)}{x_2 - x_1} \tag{7.1}$$

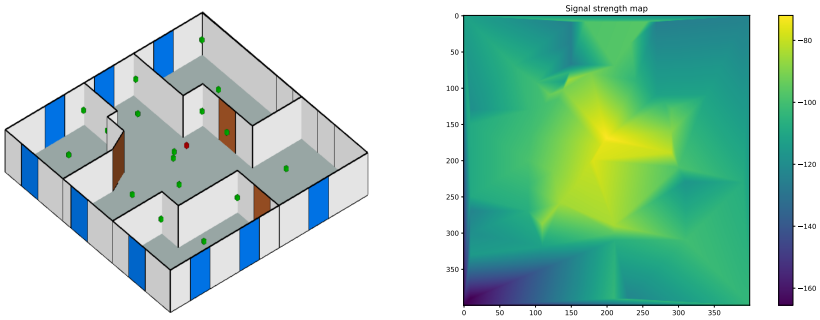


Figure 7.11: Example of a complex indoor environment with Tx antenna (red) and Rx antennas (green) and the corresponding signal strength map

2. **Simulating indoor environment** The maximum sensing range of electromagnetic sensors available in the market is 10 meters, however, in practical situations, this range is significantly smaller due to the presence of obstacles. For that, we have developed indoor architectures with dimensions of $20 \times 20m^2$, assuming the sensors are placed at their centers. This design enables us to evaluate the sensor's maximum coverage area while accounting for potential obstacles. For this dataset, we have created two types of environments: basic indoor environments and complex indoor environments. Basic indoors serve the purpose of helping the model learn fundamental patterns, physics, and wave behavior when interacting with simple shapes. Additionally, basic environments involve the use of a single construction material, allowing the model to understand how different materials (such as glass, plasterboard, concrete, brick, and wood) affect electromagnetic detection signals. This facilitates learning the distortion patterns associated with each construction material. On the other hand, complex indoor environments are designed to simulate real-world scenarios. They feature diverse architectural layouts and incorporate various construction materials. Including these complex architectures in the dataset enables the model to learn the behavior of electromagnetic waves in challenging and intricate situations. This provides a more comprehensive understanding of wave propagation in realistic indoor settings. Fig. 7.12 depicts an instance of a basic indoor environment with various construction materials, while Figure 7.11 illustrates an example of a complex indoor environment incorporating four distinct construction materials.

   For each indoor architecture, we meticulously configured both the Tx antenna and a set

of Rx antennas, with the count of Rx antennas contingent upon the particular scenario. Subsequently, a two-stage simulation was executed for each scenario, involving the comprehensive reconfiguration of all antennas. The outcome of this process consisted of the generation of detailed received signal strength maps, specifically illustrating the reflections of detection waves across the entirety of the scenarios considered.

For this dataset, we generated 13 basic indoor environments. In each one, we varied the construction materials among plasterboard, wood, concrete, glass, and brick, which are commonly used materials in indoor buildings. Additionally, we created 23 complex indoor environments with diverse architectural layouts. For each indoor architecture, we configured a Tx antenna and a set of Rx antennas. The number of Rx antennas varied depending on the architectural layout. Furthermore, we conducted a two-stage simulation for each scenario, wherein we reconfigured all antennas. Subsequently, we generated received signal strength maps to detect reflected waves for all scenarios.

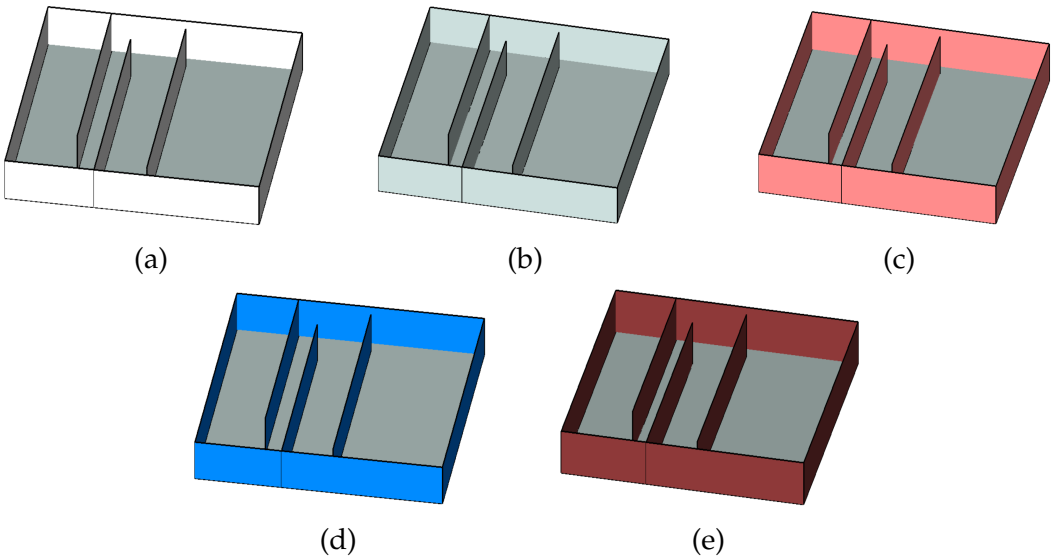


Figure 7.12: Example of a basic indoor environment with different construction materials: (a) plasterboard, (b) concrete, (c) wood, (d) glass, and (e) brick.

3. **Data augmentation** Simulating new scenarios is a time-intensive task, requiring the preparation of the environment, configuration of transmitting and receiving antennas, and the two-stage execution of ray tracing. This process makes generating a significant amount of input data challenging and time-consuming. To address this, we have explored data augmentation as a means to increase the dataset size without the need for additional simulation scenarios. This augmentation technique involves introducing variations through the rotation and flipping of existing input data. Apart from expand-

ing the dataset, this method makes the model invariant to rotations and flips which is a desirable characteristic for signal propagation as discussed in [199]. Depending on the indoor environment, various operations can be applied, including 90-degree rotation, 180-degree rotation, 270-degree rotation, left flipping, downward flipping, left flipping followed by 90-degree rotation, and downward flipping followed by a 90-degree rotation. Fig 7.13 represents an example of applying the aforementioned operations on a complex indoor scenario.

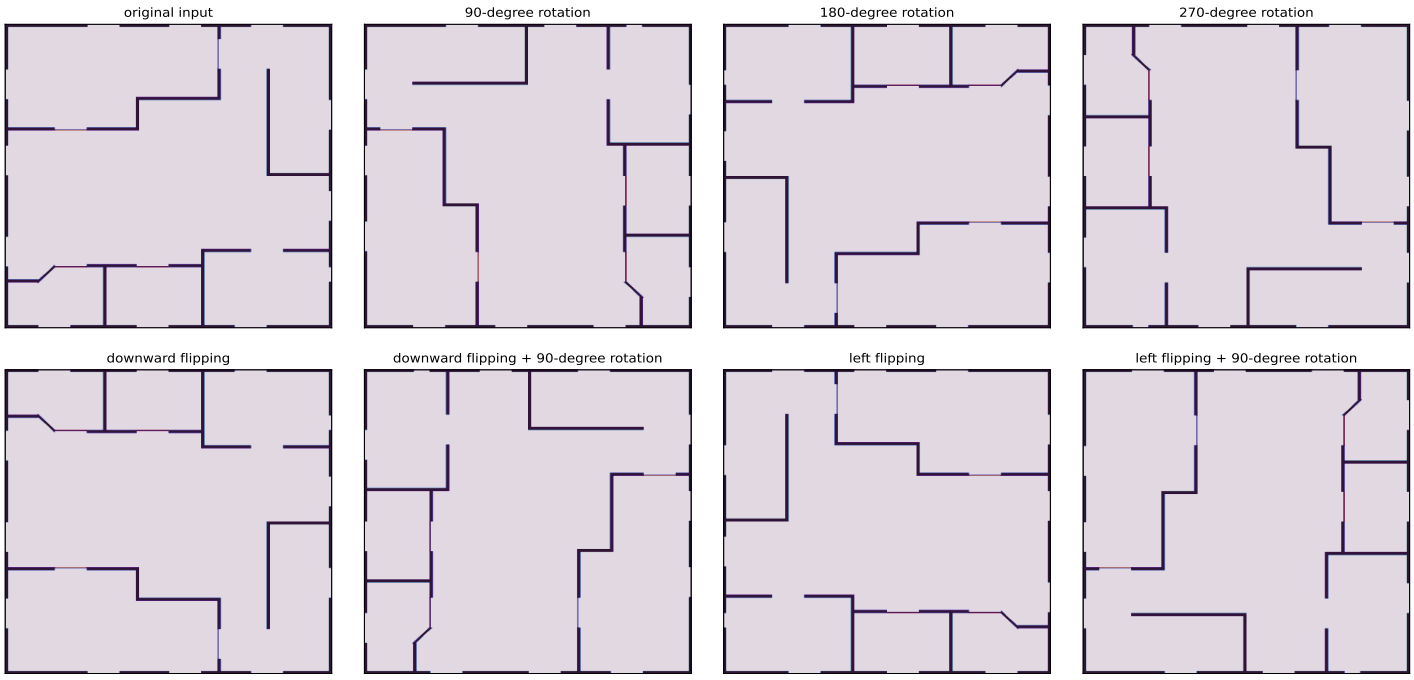


Figure 7.13: Example of applying rotation and flipping to an input indoor environment for data augmentation.

7.4.2 Model hyperparameters and evaluation metrics

Our sensing model is implemented in Python 3.7.8 using TensorFlow (version 2.11.0) and Keras (version 2.11.0). For training, we applied the Adam optimization algorithm with a learning rate set at 0.0005. We conducted training over 250 epochs, using a batch size of 4. Batch normalization is performed after each standard convolution to normalize feature maps and facilitate the convergence of the model. We utilized 390 samples for training the model, 61 samples for evaluation, and 32 samples for testing. Since the problem addressed here is a regression problem, we utilized RMSE (Root Mean Squared Error) as the loss function, as depicted in Eq. 7.2

$$\text{RMSE} = \sqrt{\frac{1}{H \times W \times C} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} \left(Ypred_{hw}^{(c)} - Ytrue_{hw}^{(c)} \right)^2} \tag{7.2}$$

Where $Ytrue$ represents the tensor with true values and $Ypred$ is the tensor with predicted values. H and W represent the height and the width of the matrices, respectively. C is the number of channels, $Ypred_{hw}^{(c)}$ and $Ytrue_{hw}^{(c)}$ are the values of the predicted and true tensors at height h, width w, and channel c respectively. For error metrics, we utilized the Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) metrics, represented by equations Eq. 7.3 and Eq. 7.4, respectively.

$$\text{MAE} = \frac{1}{H \times W \times C} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} |Ytrue_{hw}^{(c)} - Ypred_{hw}^{(c)}| \tag{7.3}$$

$$\text{MAPE} = \frac{100\%}{H \times W \times C} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} \left| \frac{Ytrue_{hw}^{(c)} - Ypred_{hw}^{(c)}}{Ytrue_{hw}^{(c)}} \right| \tag{7.4}$$

7.4.3 Generalization for new indoor environments

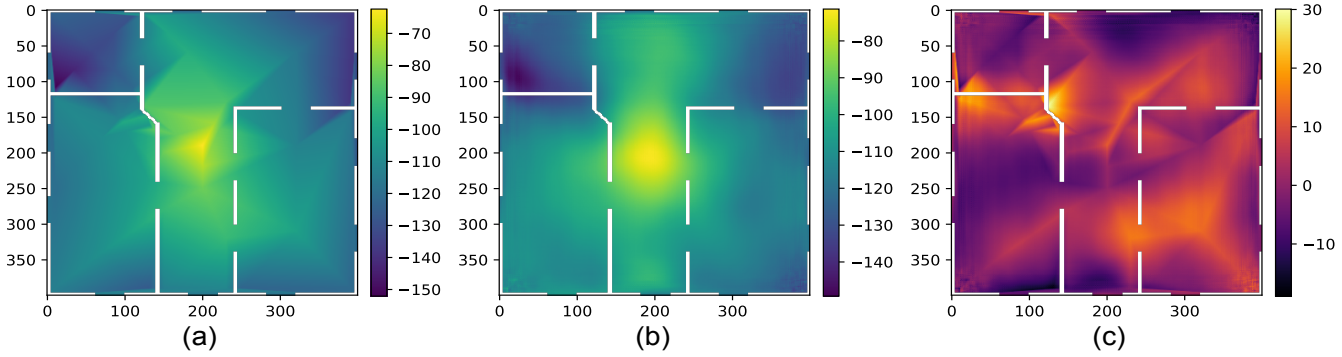
Despite the limited number of samples used for training, the model managed to achieve very satisfactory results. When evaluating the MAE metric, the model yielded a value of 6.3502 dB for the evaluation data and 6.0641 dB for unseen data. Similarly, for the MAPE error, the model exhibited values of 5.9371 dB and 5.4839 dB for the evaluation and unseen data, respectively. These errors are relatively small when considering the complexity of the addressed multi-label regression task and the wide range of output data, which spans from -70 dB to -180 dB. This indicates that the model is capable of accurately estimating signal strength across a broad spectrum of signal values. Such performance highlights also the effectiveness of the model in handling new complex indoor environments with heterogeneous obstacles.

To further assess the performance of our sensing model, we applied it to determine the detection zone of an electromagnetic sensor deployed in a new, unseen indoor setting. The sensing model lacks any previous data about the signal strength in this area and endeavors to estimate it using the trained model. Figure 7.14 illustrates four randomly

selected unseen indoor environments used for testing. In the first column (a), we present the ground truth detection signal strength map generated through ray tracing. Column (b) depicts the estimated detection signal strength for the same environments, while column (c) showcases the error, indicating disparities between the ground truth and predicted signal strengths at each location in the new indoor environments. Each signal map has been generated by the sensing model in less then a second. An important observation is that the model was trained using signal maps generated with linear interpolation, which are characterized by abrupt signal strength changes in some areas. However, the proposed sensing model produces smoother maps compared to the ground truth maps, showcasing a more gradual transition. This smoothness better captures the propagation characteristics of detection signals. In nearby regions of the sensor, the model predicts values close to the ground truth, showing minimal error. Conversely, the largest errors are observed in distant locations, which already lie beyond the sensor's detection range due to obstructing obstacles. Moreover, it is observed from these figures that the attenuation of the signal when encountering heterogeneous obstacles aligns with the attenuation pattern observed in the ground truth signal map. Areas partially obscured by surrounding obstacles display reduced signal strength in both ground truth and predicted signal maps. This confirms the capability of our sensing model to infer the influence of obstacles on detection signals. Furthermore, the predicted detection signal values closely align with the distribution of obstacles, particularly within a 5-meter square around the sensor. This results in minimal error values ranging between -10 and 10 within this square (figures (c)).

7.4.4 Comparison with existing sensing models

Our proposed sensing model provides a more precise assessment of electromagnetic sensor detection zones in indoor areas compared to existing models in the literature.



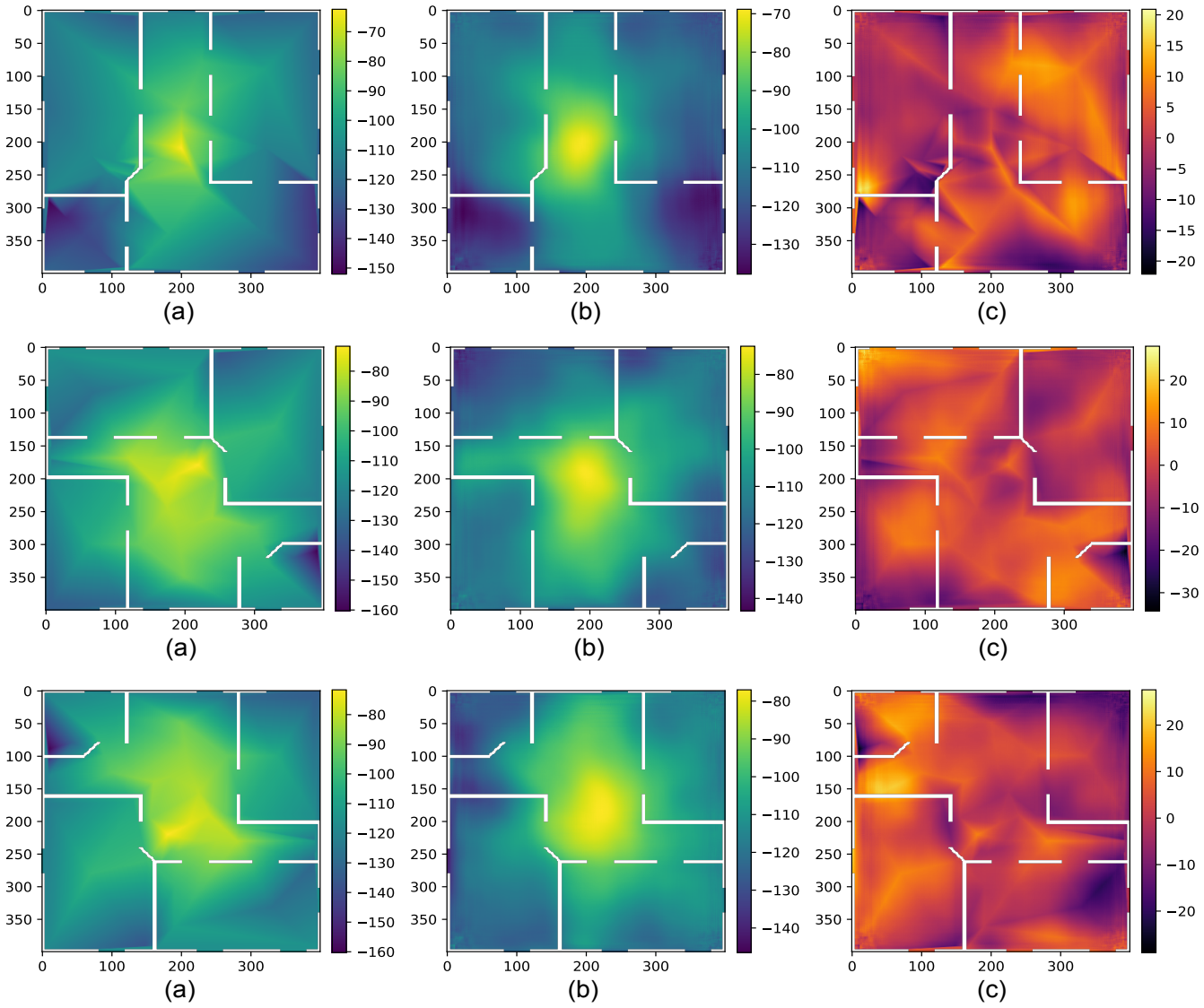


Figure 7.14: Examples of signal strength maps for unseen indoor environments: (a) map simulated by ray tracing, (b) map predicted by our sensing model, and (c) error matrix.

This superiority stems from its ability to take into account the propagation behaviors of detection signals, enabling the estimation of signal attenuation and subsequent computation of signal strength. Unlike existing models, which typically model signal attenuation as a random variable (Shadow fading model), our model does not assume a uniform sensing ability in all directions. Instead, it factors in the distribution of obstacles, accounting for their various shapes and sizes. Furthermore, a significant advantage of our model over existing ones lies in its capacity to incorporate the heterogeneity of surrounding building obstacles, leading to a more precise estimation. The main characteristics of our sensing model, as compared to existing models, are summarized in Table 7.1.

Table 7.1: Comparison between our proposed sensing model and existing models.

Sensing model	Distance	Heterogeneity of obstacles	sensors characteristics	Signal Propagation	Non-uniform sensing ability
Deterministic model	✓	X	X	X	X
Sigmoid model	✓	X	✓	X	X
Attenuated Disk model	✓	X	X	X	X
Exponential Model	✓	X	X	X	X
Probabilistic model with noise	✓	X	X	X	X
Shadow fading model	✓	X	X	X	✓
Elfes model	✓	X	✓	X	✓
Our proposed model	✓	✓	✓	✓	✓

7.5 Integration of the proposed sensing model with the overall WSN deployment approach

The developed sensing model can be seamlessly incorporated into the proposed deployment approaches to measure network coverage, defined as the total number of target points covered by all sensors. The deployment approaches generate WSN deployment schemes, which are encoded by binary vectors through the assignment of sensors to designated deployment zones as explained in Section 4.3. To calculate the coverage for a given WSN topology, we initiate the process by examining the binary vector to identify the positions of all sensors, as determined by the optimization method. Subsequently, for each sensor, the following steps are performed:

1. Extract all building obstacles surrounding the sensor from the BIM database. The sensor position is identified by its coordinates within the building. Therefore, to extract the building obstacles and their characteristics surrounding the sensor, we begin by computing the coordinates of the vertices of the square that delimits the maximum detection zone. The Euclidean distance between the sensor and each edge is set at 10m, as electromagnetic sensors for indoor environments cannot go beyond this distance, as explained earlier. Once the maximum detection zone is identified, a spatial query is conducted on the BIM database to extract all building obstacles and their characteristics that fall within this square.

2. Generate inputs representing the surrounding area: The extracted building data contains the layout of the surrounding area, including position, dimensions, and construction materials. Since this data is extracted from an IFC file, it needs to be processed by

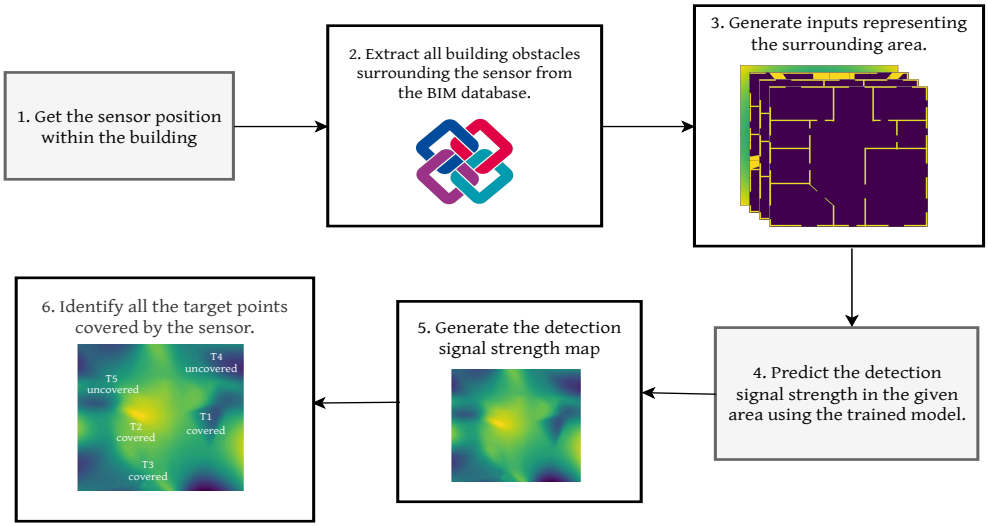


Figure 7.15: Example of applying rotation and flipping to an input indoor environment for data augmentation.

the Geometric Data Interpreter and Geometric Data Organizer modules presented in Section 4.2.3 to convert it into exploitable information depicting the indoor area. From this information, we generate the corresponding input tensor channels (permittivity channel, conductivity channel, distance channel, and LoS channel) for the given area, as detailed in Section 7.3.1.

3. Generate the detection signal strength map through the application of the trained model: The tensor representing the sensor's maximum detection zone is sent to the trained model for predicting signal strength across all locations within this designated area. The resulting output from the model is the signal strength map, represented by a 400 $\times$ 400 matrix, as explained in Section 7.3.2.

4. Identify all target points within the sensor's coverage: Detecting all target points within the sensor's coverage involves recognizing areas where the signal strength surpasses the sensor's sensitivity threshold. If the signal strength is above this threshold, the corresponding zones are considered within the sensor's coverage; otherwise, they are deemed beyond its detection zone. Consequently, a target point is deemed covered by the sensor if it resides within the first set of zones; otherwise, it is not included in the coverage assessment.

After executing this process for all sensors in the WSN topology, coverage is assessed by considering the total number of target points covered by all sensors. Fig 7.15 illustrates the steps of integrating the proposed sensing model with the overall WSN deployment

approach based on the BIM database. In this example, target points $T1$, $T2$, and $T3$ are considered covered by the given sensor, while target points $T4$ and $T5$ are uncovered.

7.6 Conclusion

Electromagnetic sensors stand out as the most commonly utilized non-contact sensors in smart buildings. They operate by emitting electromagnetic waves across the deployment area and analyzing the reflected detection signals to extract useful information, such as detecting the presence of people and recognizing their activities. Primarily, electromagnetic sensors encompass occupancy sensors and motion sensors, both of which play a pivotal role in numerous smart building applications. Thus, the accurate estimation of the detection zone of electromagnetic sensors significantly impacts the overall performance of smart buildings.

Existing sensing models in literature for assessing the detection zone of these sensors exhibit limitations, leading to inaccuracies in network coverage. Therefore, in this chapter, we propose a new sensing model tailored for electromagnetic sensors within smart buildings. This model takes into account the characteristics of the surrounding area where the sensor is deployed, including the indoor layout and the construction materials of building obstacles, along with their thickness and dimensions. Such considerations enable our sensing model to better estimate the propagation of detection signals and their strength. The output of our sensing model is a signal map, where each cell denotes the strength of the detection signal reflected from its corresponding position in the deployment area. Positions with signal strengths above the sensor sensitivity threshold are considered covered, while those below are deemed uncovered. Our model has demonstrated its effectiveness in predicting detection signal strength across various unseen indoor areas. Notably, the predicted maps replicate the signal behaviors simulated by ray tracing, but with significantly reduced computational time. The predicted maps are generated in less than one second, whereas ray tracing takes several minutes (up to 20 minutes) to generate the signal map for the same scenario. This chapter explains the data generation process and the method for integrating this sensing model with BIM and WSN deployment approaches. Furthermore, it highlights the advantages of our sensing model compared to existing models in the literature.

CONCLUSION AND RESEARCH PERSPECTIVES

Conclusion

In this thesis, we have tackled the problem of WSN deployment in smart buildings. The objective is to propose sensor network topologies that maximize coverage and minimize the number of deployed sensor nodes while ensuring the connectivity constraint. In the first stage, we formulated the problem as a single-objective combinatorial problem where the fitness function aggregates both coverage and cost with the same weight. This phase comprised two studies. The first study compared seven well-known metaheuristics for WSN deployment. We comprehensively tested their performance across ten indoor architectures of varying sizes. Based on the obtained results, we propose an enhanced binary grey wolf optimizer incorporating a Steiner tree-based heuristic. The objective of this heuristic is to identify a subset of sensors within a WSN topology that preserves the same coverage rate and satisfies the connectivity constraint. Consequently, sensors not included in the selected subset are eliminated from the final topology, thereby reducing the deployment cost. This heuristic allows a considerable improvement in the convergence of the algorithm and produces the best solutions for all the considered scenarios.

Considering the multi-objective nature of the problem, we investigated multi-objective metaheuristics through a comparative study involving evolutionary and swarm-based algorithms. The results indicate the effectiveness of MOGWO and NSGA-II in addressing the problem under evaluation budget constraints. In the subsequent research phase, we introduced a hybrid multi-objective approach, LHNSGA-II, which integrates MOGWO with NSGA-II by incorporating the MOGWO archive into the initial population of NSGA-II to enhance its exploitation process. Additionally, a local search was developed to re-

fine generated topologies in two stages. The initial stage involves relocating sensors to neighboring deployment zones to identify the best first zone that enhances coverage and meets the connectivity constraint. The second stage focuses on eliminating dominated sensors that do not contribute to network connectivity. All optimization algorithms incorporate a proposed connectivity repair heuristic. This heuristic aims to link disjoint sensor sets in unconnected topologies generated during the optimization process by deploying a minimal number of additional nodes. These proposed hybrid approaches highlighted the advantages of metaheuristic hybridization and the incorporation of problem-specific heuristics to improve the quality of the solution.

To solve the problem of target area modeling, we have developed a BIM-centered system. This system comprises two key components: the data extraction component and the deployment zone modeling. The data extraction component functions as a parser that extracts essential information from the IFC file describing the BIM model, particularly the entities representing the building layout and obstacles. These extracted IFC entities are processed to facilitate their utilization in the second component, which is the deployment zone modeling. In this stage, the deployment zone modeling component partitions each floor into a set of zones where sensors can be positioned.

In the final contribution, we introduced a novel sensing model. This model is designed for electromagnetic sensors commonly used in smart buildings to collect data related to human presence detection, people counting and tracking, as well as the recognition of people activities. It is based on fully convolutional networks and takes into consideration the deployment environment and its influence on the detection signals of electromagnetic sensors. The model produces a signal strength map illustrating the covered and uncovered zones by a given sensor. The input to the model consists of four channels describing the sensor's surrounding environment. The first two channels are permittivity and conductivity channels. The combination of these channels provides essential information on the distribution of building obstacles, including their thickness and construction material. The third channel represents the distances between the sensor and other zones, this enables the model to learn the influence of distance on signal strength. Lastly, the line of sight channel which depicts all zones reachable by the sensor detection signals without encountering obstacles. The sensing model is based on the SDU-net which is a variation of the U-net architecture that incorporates atrous convolutions to augment the receptive field. Specifically designed for semantic segmentation, this architecture enables the assignment of detection signal values to each zone within the designated deployment area.

Research perspectives

Both the sensing model and WSN deployment approach can be extended for further improvement by considering additional parameters as follows:

- **WSN deployment approach:** The first enhancement for the deployment approach involves integrating the proposed sensing model, as detailed in Section 7.5. To further improve the applicability of our deployment approach, it can be integrated into a real Building BIM tool. For instance, the approach could be incorporated as a plugin in software like Revit. The deployment process could then be executed as an internal script accessing the IFC file, which describes the BIM database, to compute deployment zones prior to optimization. Additionally, while this thesis focuses on the deployment of sensors on a single floor simultaneously, it is crucial to consider deploying the sensor network across entire buildings. This entails deploying some sink nodes strategically to reduce the number of required sensors and ensure connectivity throughout the building. Deploying WSNs in large buildings with multiple floors and complex layouts, along with heterogeneous obstacles, is a time-consuming task. Therefore, optimizing the execution time of optimization techniques becomes essential. One way to address this is by incorporating surrogate models to estimate the network coverage of certain topologies within the population. This approach avoids the need to compute the detection zone for each sensor individually, thus reducing overall execution time.

- **Sensing Model for electromagnetic sensors:**

  An interesting perspective of this component is to share it with the research community as a readily deployable black box solution that can be easily utilized by deployment approaches to assess WSNs coverage. However, due to the complexity of dataset generation and the significant time required to reconfigure antennas and execute ray tracing in two stages for each indoor scenario, it was impossible to generate a large amount of data to train our model. Therefore, it would be beneficial to augment the number of samples and consider additional receiver antennas for a more refined signal strength map. This model could also be generalized to other electromagnetic sensor types with different frequencies. This diversity of sensors could be incorporated as frequency channels to each input. Furthermore, a crucial consideration involves evaluating the influence of mobile obstacles on the signal strength map produced by the sensing model. This evaluation requires the inclusion of novel inputs derived from simulating mobile obstacles across varied indoor

environments. Such an approach enhances the sensing model's adaptability, particularly in response to dynamic scenarios. Additionally, it would be advantageous to assess the accuracy of the model using testbeds, comparing the predicted detection zone with the ground truth.

BIBLIOGRAPHY

[1] Umit Isikdag. "Enhanced building information models". In: *SpringerBriefs in Computer Science* (2015), pp. 577–590.

[2] Antony Guinard, Alan McGibney, and Dirk Pesch. "A wireless sensor network design tool to support building energy management". In: *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. 2009, pp. 25–30.

[3] Jianchao Zhang, Boon-Chong Seet, and Tek Tjing Lie. "Building information modelling for smart built environments". In: *Buildings* 5.1 (2015), pp. 100–115.

[4] Dina S Deif and Yasser Gadallah. "Classification of wireless sensor networks deployment techniques". In: *IEEE Communications Surveys & Tutorials* 16.2 (2013), pp. 834–855.

[5] Khaoula Zaimen et al. "A survey of artificial intelligence based wsns deployment techniques and related objectives modeling". In: *IEEE Access* (2022).

[6] Zesong Fei et al. "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems". In: *IEEE Communications Surveys & Tutorials* 19.1 (2016), pp. 550–586.

[7] John H Holland. "Genetic algorithms". In: *Scientific american* 267.1 (1992), pp. 66–73.

[8] Rainer Storn and Kenneth Price. "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces". In: *Journal of global optimization* 11.4 (1997), pp. 341–359.

[9] Xin-She Yang and Suash Deb. "Cuckoo search via Lévy flights". In: *2009 World congress on nature & biologically inspired computing (NaBIC)*. Ieee. 2009, pp. 210–214.

[10] James Kennedy and Russell Eberhart. "Particle swarm optimization". In: *Proceedings of ICNN'95-international conference on neural networks*. Vol. 4. IEEE. 1995, pp. 1942–1948.

[11] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer". In: *Advances in engineering software* 69 (2014), pp. 46–61.

[12] Xin-She Yang. "A new metaheuristic bat-inspired algorithm". In: *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, 2010, pp. 65–74.

[13] Zong Woo Geem, Joong Hoon Kim, and Gobichettipalayam Vasudevan Loganathan. "A new heuristic optimization algorithm: harmony search". In: *simulation* 76.2 (2001), pp. 60–68.

[14] Salah Eddine Bouzid. "Optimisation multicritères des performances de réseau d'objets communicants par méta-heuristiques hybrides et apprentissage par renforcement". PhD thesis. Le Mans, 2020.

[15] M Santamouris and K Vasilakopoulou. "Present and future energy consumption of buildings: Challenges and opportunities towards decarbonisation". In: *e-Prime-Advances in Electrical Engineering, Electronics and Energy* 1 (2021), p. 100002.

[16] Mohamed Amin Benatia. "Optimisation multi-objectives d'une infrastructure réseau dédiée aux bâtiments intelligents". PhD thesis. Rouen, INSA, 2016.

[17] Marko Djurasevic et al. "A survey of metaheuristic algorithms for the design of cryptographic Boolean functions". In: *Cryptography and Communications* 15.6 (2023), pp. 1171–1197.

[18] Ian F Akyildiz et al. "Wireless sensor networks: a survey". In: *Computer networks* 38.4 (2002), pp. 393–422.

[19] Winncy Y Du. *Resistive, capacitive, inductive, and magnetic sensor technologies*. CRC Press, 2014.

[20] Riham Elhabyan, Wei Shi, and Marc St-Hilaire. "Coverage protocols for wireless sensor networks: Review and future directions". In: *Journal of Communications and Networks* 21.1 (2019), pp. 45–60.

[21] Mohammad Abdul Matin and MM Islam. "Overview of wireless sensor network". In: *Wireless sensor networks-technology and protocols* 1.3 (2012).

[22] Alex H Buckman, Martin Mayfield, and Stephen BM Beck. "What is a smart building?" In: *Smart and Sustainable Built Environment* 3.2 (2014), pp. 92–109.

[23] Joud Al Dakheel et al. "Smart buildings features and key performance indicators: A review". In: *Sustainable Cities and Society* 61 (2020), p. 102328.

[24] Wael Alsafery, Omer Rana, and Charith Perera. "Sensing within smart buildings: A survey". In: *ACM Computing Surveys* 55.13s (2023), pp. 1–35.

[25] Anxing Shan, Xianghua Xu, and Zongmao Cheng. "Target Coverage in Wireless Sensor Networks with Probabilistic Sensors". In: *Sensors* 16.9 (2016). ISSN: 1424-8220. DOI: 10.3390/s16091372.

[26] Edoardo Amaldi et al. "Design of wireless sensor networks for mobile target detection". In: *IEEE/ACM transactions on networking* 20.3 (2011), pp. 784–797.

[27] Manju Chaudhary and Arun K Pujari. "Q-coverage problem in wireless sensor networks". In: *International Conference on Distributed Computing and Networking*. Springer. 2009, pp. 325–330.

[28] Mo Li et al. "Sweep coverage with mobile sensors". In: *IEEE Transactions on Mobile Computing* 10.11 (2011), pp. 1534–1545.

[29] Peng Huang et al. "GA-based sweep coverage scheme in WSN". In: *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*. IEEE. 2016, pp. 254–259.

[30] Zixiong Nie and Hongwei Du. "An approximation algorithm for General Energy Restricted Sweep Coverage problem". In: *Theoretical Computer Science* 864 (2021), pp. 70–79.

[31] Benyuan Liu et al. "Strong barrier coverage of wireless sensor networks". In: *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*. 2008, pp. 411–420.

[32] J Amutha, Sandeep Sharma, and Jaiprakash Nagar. "WSN strategies based on sensors, deployment, sensing models, coverage and energy efficiency: Review, approaches and open issues". In: *Wireless Personal Communications* 111.2 (2020), pp. 1089–1115.

[33] Junbin Liang, Ming Liu, and Xiaoyan Kui. "A survey of coverage problems in wireless sensor networks". In: *Sensors & Transducers* 163.1 (2014), pp. 240–246.

[34] Sanay Abdollahzadeh and Nima Jafari Navimipour. "Deployment strategies in the wireless sensor network: A comprehensive review". In: *Computer Communications* 91 (2016), pp. 1–16.

[35] Ridha Soua and Pascale Minet. "A survey on energy efficient techniques in wireless sensor networks". In: *2011 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2011)*. IEEE. 2011, pp. 1–9.

[36] Imen Ben Arbi, Faouzi Derbel, and Florian Strakosch. "Forecasting methods to reduce energy consumption in WSN". In: *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. IEEE. 2017, pp. 1–6.

[37] Prachi Maheshwari, Ajay K Sharma, and Karan Verma. "Energy efficient cluster based routing protocol for WSN using butterfly optimization algorithm and ant colony optimization". In: *Ad Hoc Networks* 110 (2021), p. 102317. ISSN: 1570-8705.

[38] Reeta Bhardwaj and Dinesh Kumar. "MOFPL: Multi-objective fractional particle lion algorithm for the energy aware routing in the WSN". In: *Pervasive and Mobile Computing* 58 (2019), p. 101029.

[39] A. Vinitha, M.S.S. Rukmini, and Dhirajsunehra. "Secure and Energy Aware Multi-hop Routing protocol in WSN using Taylor-based Hybrid Optimization Algorithm". In: *Journal of King Saud University - Computer and Information Sciences* 34 (Nov. 2019). DOI: 10.1016/j.jksuci.2019.11.009.

[40] Marlon Jeske, Valério Rosset, and Mariá CV Nascimento. "Determining the trade-offs between data delivery and energy consumption in large-scale WSNs by multi-objective evolutionary optimization". In: *Computer Networks* 179 (2020), p. 107347.

[41] Giuseppe Anastasi et al. "Energy conservation in wireless sensor networks: A survey". In: *Ad hoc networks* 7.3 (2009), pp. 537–568.

[42] Ramesh Rajagopalan et al. "Multi-objective mobile agent routing in wireless sensor networks". In: *2005 IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE. 2005, pp. 1730–1737.

[43] Yang Wang et al. "Coverage problem with uncertain properties in wireless sensor networks: A survey". In: *Computer Networks* 123 (2017), pp. 200–232.

[44] Srabani Kundu, Nabanita Das, and Dibakar Saha. "A Realistic Sensing Model for Event Area Estimation in Wireless Sensor Networks". In: *Progress in Advanced Computing and Intelligent Engineering*. Springer, 2021, pp. 244–256.

[45] Sushil Kumar and DK Lobiyal. "Sensing coverage prediction for wireless sensor networks in shadowed and multipath environment". In: *The Scientific World Journal* 2013 (2013), p. 565419.

[46] J Vidhya et al. "Enhancing Network Coverage Using Sensing Models in Wireless Sensor Network". In: *Journal of Physics: Conference Series*. Vol. 1717. IOP Publishing. 2021, p. 012062.

[47] Vahab Akbarzadeh et al. "Probabilistic sensing model for sensor placement optimization based on line-of-sight coverage". In: *IEEE Transactions on Instrumentation and Measurement* 62.2 (2012), pp. 293–303.

[48] Bang Wang. "Coverage problems in sensor networks: A survey". In: *ACM Computing Surveys (CSUR)* 43.4 (2011), pp. 1–53.

[49] Huynh Thi Thanh Binh et al. "Efficient approximation approaches to minimal exposure path problem in probabilistic coverage model for wireless sensor networks". In: *Applied Soft Computing* 76 (2019), pp. 726–743.

[50] Ashraf Hossain, Prabir Kumar Biswas, and Saswat Chakrabarti. "Sensing models and its impact on network coverage in wireless sensor network". In: *2008 IEEE Region 10 and the Third international Conference on Industrial and Information Systems*. IEEE. 2008, pp. 1–5.

[51] Nitika Rai and Rohin Daruwala. "Node density optimisation using composite probabilistic sensing model in wireless sensor networks". In: *IET Wireless Sensor Systems* 9.4 (2019), pp. 181–190.

[52] Govind Sati and Sonika Singh. "A review on outdoor propagation models in radio communication". In: *International Journal of Computer Engineering & Science* 4.2 (2014), pp. 64–68.

[53] Zahera Naseem, Iram Nausheen, and Zahwa Mirza. "Propagation models for wireless communication system". In: *nternational Journal of Research in Engineering and Technology* 5.01 (2018), pp. 237–242.

[54] Meiling Luo. "Indoor radio propagation modeling for system performance prediction". PhD thesis. INSA de Lyon, 2013.

[55] Josip Milanović, Snježana Rimac-Drlje, and Ivo Majerski. "Radio wave propagation mechanisms and empirical models for fixed wireless access systems". In: *Tehnički vjesnik* 17.1 (2010), pp. 43–53.

[56] Ayoub Saad, Mustapha Reda Senouci, and Oussama Benyattou. "Toward a realistic approach for the deployment of 3D Wireless Sensor Networks". In: *IEEE Transactions on Mobile Computing* (2020).

[57] Bin Cao et al. "Deployment optimization for 3D industrial wireless sensor networks based on particle swarm optimizers with distributed parallelism". In: *Journal of Network and Computer Applications* 103 (2018), pp. 225–238.

[58] Bin Cao et al. "3D terrain multiobjective deployment optimization of heterogeneous directional sensor networks in security monitoring". In: *IEEE Transactions on Big Data* 5.4 (2017), pp. 495–505.

[59] Xiangyu Yu et al. "A faster convergence artificial bee colony algorithm in sensor deployment for wireless sensor networks". In: *International Journal of Distributed Sensor Networks* 9.10 (2013), p. 497264.

[60] Marc T Kouakou et al. "Cost-efficient sensor deployment in indoor space with obstacles". In: *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE. 2012, pp. 1–9.

[61] Zhendong Wang and Huamao Xie. "Wireless Sensor Network Deployment of 3D Surface Based on Enhanced Grey Wolf Optimizer". In: *IEEE Access* 8 (2020), pp. 57229–57251.

[62] Salah Eddine Bouzid et al. "MOONGA: multi-objective optimization of wireless network approach based on genetic algorithm". In: *IEEE Access* 8 (2020), pp. 105793–105814.

[63] Salah Eddine Bouzid et al. "Wireless sensor network deployment optimisation based on coverage, connectivity and cost metrics". In: *International Journal of Sensor Networks* 33.4 (2020), pp. 224–238.

[64] Mohamed Amin Benatia et al. "Multi-objective WSN deployment using genetic algorithms under cost, coverage, and connectivity constraints". In: *Wireless Personal Communications* 94.4 (2017), pp. 2739–2768.

[65] Khaoula Zaimen et al. "A overview on WSN deployment and a novel conceptual BIM-based approach in smart buildings". In: *2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. IEEE. 2020, pp. 1–6.

[66] Wendi Fu et al. "WSN Deployment Strategy for Real 3D Terrain Coverage Based on Greedy Algorithm with DEM Probability Coverage Model". In: *Electronics* 10.16 (2021), p. 2028.

[67] Nguyen Tam et al. “Multi-objective teaching–learning evolutionary algorithm for enhancing sensor network coverage and lifetime”. In: *Engineering Applications of Artificial Intelligence* 108 (2022), p. 104554.

[68] Meysam Argany et al. “Optimization of wireless sensor networks deployment based on probabilistic sensing models in a complex environment”. In: *Journal of Sensor and Actuator Networks* 7.2 (2018), p. 20.

[69] Khaoula Zaimen et al. “Coverage Maximization in WSN Deployment Using Particle Swarm Optimization with Voronoi Diagram”. In: *International Conference on Model and Data Engineering*. Springer. 2021, pp. 88–100.

[70] Ali Metiaf and Qianhong Wu. “Particle Swarm Optimization Based Deployment for WSN with the Existence of Obstacles”. In: *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE. 2019, pp. 614–618.

[71] Mohammed Abo-Zahhad et al. “A centralized immune-Voronoi deployment algorithm for coverage maximization and energy conservation in mobile wireless sensor networks”. In: *Information Fusion* 30 (2016), pp. 36–51.

[72] Nejah Nasri, Sami Mnasri, and Thierry Val. “3D node deployment strategies prediction in wireless sensors network”. In: *International Journal of Electronics* 107.5 (2020), pp. 808–838.

[73] Usman Mansoor and Habib M Ammari. “Coverage and connectivity in 3D wireless sensor networks”. In: *The art of wireless sensor networks*. Springer, 2014, pp. 273–324.

[74] Chun Kit Ng et al. “A smart bat algorithm for wireless sensor network deployment in 3-D environment”. In: *IEEE Communications Letters* 22.10 (2018), pp. 2120–2123.

[75] Hashim A Hashim, Babajide Odunitan Ayinde, and Mohamed A Abido. “Optimal placement of relay nodes in wireless sensor network using artificial bee colony algorithm”. In: *Journal of Network and Computer Applications* 64 (2016), pp. 239–248.

[76] Deepika Sharma and Vrinda Gupta. “Modeling 3D WSN to maximize coverage using harmony search scheme”. In: *Advanced Engineering Optimization Through Intelligent Techniques*. Springer, 2020, pp. 95–111.

[77] Vahab Akbarzadeh et al. “Efficient sensor placement optimization using gradient descent and probabilistic coverage”. In: *Sensors* 14.8 (2014), pp. 15525–15552.

[78] Hui Wu et al. “Sensor placement optimization for critical-grid coverage problem of indoor positioning”. In: *International Journal of Distributed Sensor Networks* 16.12 (2020), p. 1550147720979922.

[79] Joon-Yong Lee, Joon-Hong Seok, and Ju-Jang Lee. "Multiobjective optimization approach for sensor arrangement in a complex indoor environment". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.2 (2011), pp. 174–186.

[80] Sami Mnasri et al. "Improved many-objective optimization algorithms for the 3D indoor deployment problem". In: *Arabian Journal for Science and Engineering* 44.4 (2019), pp. 3883–3904.

[81] Ali Afghantoloee and Mir Abolfazl Mostafavi. "A Local 3D Voronoi-Based Optimization Method for Sensor Network Deployment in Complex Indoor Environments". In: *Sensors* 21.23 (2021), p. 8011.

[82] Ali Jameel Al-Mousawi. "Evolutionary intelligence in wireless sensor network: routing, clustering, localization and coverage". In: *Wireless Networks* 26.8 (2020), pp. 5595–5621.

[83] Adam Slowik and Halina Kwasnicka. "Evolutionary algorithms and their applications to engineering problems". In: *Neural Computing and Applications* 32.16 (2020), pp. 12363–12379.

[84] Kangshun Li et al. "Optimised placement of wireless sensor networks by evolutionary algorithm". In: *International Journal of Computational Science and Engineering* 15.1-2 (2017), pp. 74–86.

[85] Huynh Thi Thanh Binh et al. "Metaheuristics for maximization of obstacles constrained area coverage in heterogeneous wireless sensor networks". In: *Applied Soft Computing* 86 (2020), p. 105939.

[86] Daifeng Zhang and Jiliang Zhang. "Multi-species evolutionary algorithm for wireless visual sensor networks coverage optimization with changeable field of views". In: *Applied Soft Computing* 96 (2020), p. 106680.

[87] Walid Miloud Dahmane et al. "A BIM-based framework for an Optimal WSN Deployment in Smart Building". In: *2020 11th International Conference on Network of the Future (NoF)*. IEEE. 2020, pp. 110–114.

[88] Mina Khalesian and Mahmoud Reza Delavar. "Wireless sensors deployment optimization using a constrained Pareto-based multi-objective evolutionary approach". In: *Engineering Applications of Artificial Intelligence* 53 (2016), pp. 126–139.

[89] Arouna Ndam Njoya et al. "Optimization of sensor deployment using multi-objective evolutionary algorithms". In: *Journal of Reliable Intelligent Environments* 2.4 (2016), pp. 209–220.

[90] Francisco Domingo-Perez et al. “Optimization of the coverage and accuracy of an indoor positioning system with a variable number of sensors”. In: *Sensors* 16.6 (2016), p. 934.

[91] Tingli Xiang, Hongiun Wang, and Yingchun Shi. “Hybrid WSN node deployment optimization strategy based on CS algorithm”. In: *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. IEEE. 2019, pp. 621–625.

[92] Shafaq B Chaudhry et al. “Pareto-based evolutionary computational approach for wireless sensor placement”. In: *Engineering Applications of Artificial Intelligence* 24.3 (2011), pp. 409–425.

[93] Samayveer Singh et al. “Genetic algorithm-based data controlling method using IoT-enabled WSN in power grid”. In: *Soft Computing* (2022), pp. 1–19.

[94] Huynh Binh et al. “Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks”. In: *Neural computing and applications* 30.7 (2018), pp. 2305–2317.

[95] D Arivudainambi, R Pavithra, and P Kalyani. “Cuckoo search algorithm for target coverage and sensor scheduling with adjustable sensing range in wireless sensor network”. In: *Journal of Discrete Mathematical Sciences and Cryptography* (2020), pp. 1–22.

[96] Elivelton O Rangel, Daniel G Costa, and Angelo Loula. “On redundant coverage maximization in wireless visual sensor networks: Evolutionary algorithms for multi-objective optimization”. In: *Applied Soft Computing* 82 (2019), p. 105578.

[97] Juhi R Srivastava and TSB Sudarshan. “Energy-efficient cache node placement using genetic algorithm in wireless sensor networks”. In: *Soft Computing* 19.11 (2015), pp. 3145–3158.

[98] Rabun Kosar and Cem Ersoy. “Sink placement on a 3D terrain for border surveillance in wireless sensor networks”. In: *Engineering Applications of Artificial Intelligence* 25.1 (2012), pp. 82–93.

[99] Kalyanmoy Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[100] Jose M Lanza-Gutierrez and Juan A Gomez-Pulido. “Studying the multiobjective variable neighbourhood search algorithm when solving the relay node placement problem in wireless sensor networks”. In: *Soft Computing* 20.1 (2016), pp. 67–86.

[101] Nour El-Houda Benalia et al. “MoEA-DeployWSN-SB: Three variants of multi-objective evolutionary algorithms for the deployment optimization strategy of a WSN in a smart building”. In: *International Journal of Information Technology* 14.1 (2022), pp. 333–344.

[102] Arouna Ndam Njoya et al. “Optimization of sensor deployment using multi-objective evolutionary algorithms”. In: *Journal of Reliable Intelligent Environments* 2 (2016), pp. 209–220.

[103] Amrita Chakraborty and Arpan Kumar Kar. “Swarm intelligence: A review of algorithms”. In: *Nature-inspired computing and optimization* (2017), pp. 475–494.

[104] Mohd Nadhir Ab Wahab, Samia Nefti-Meziani, and Adham Atyabi. “A comprehensive review of swarm optimization algorithms”. In: *PloS one* 10.5 (2015), e0122827.

[105] Lucija Brezočnik, Iztok Fister, and Vili Podgorelec. “Swarm intelligence algorithms for feature selection: a review”. In: *Applied Sciences* 8.9 (2018), p. 1521.

[106] Yanzhi Du. “Method for the Optimal Sensor Deployment of WSNs in 3D Terrain Based on the DPSOVF Algorithm”. In: *IEEE Access* 8 (2020), pp. 140806–140821.

[107] Maryam Shakeri et al. “Performance Analysis of IoT-Based Health and Environment WSN Deployment”. In: *Sensors* 20.20 (2020), p. 5923.

[108] Wei Qi et al. “WSN Coverage Optimization Based on Two-Stage PSO”. In: *International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Springer. 2020, pp. 19–35.

[109] Yujiang Li and Jinghua Cao. “WSN node optimal deployment algorithm based on adaptive binary particle swarm optimization”. In: *ASP Transactions on Internet of Things* 1.1 (2021), pp. 1–8.

[110] Ramin Yarinezhad and Seyed Naser Hashemi. “A sensor deployment approach for target coverage problem in wireless sensor networks”. In: *Journal of Ambient Intelligence and Humanized Computing* (2020), pp. 1–16.

[111] Jian-Li Xie, Cui-Ran Li, and Guo-Yan Xu. “Optimization and Energy-saving Deployment of Nodes in Wireless Sensor Network”. In: *Journal of Computers* 31.3 (2020), pp. 142–153.

[112] Qingjian Ni et al. “An improved dynamic deployment method for wireless sensor network based on multi-swarm particle swarm optimization”. In: *Natural Computing* 16.1 (2017), pp. 5–13.

[113] Pyari Mohan Pradhan and Ganapati Panda. "Connectivity constrained wireless sensor deployment using multiobjective evolutionary algorithms and fuzzy decision making". In: *Ad Hoc Networks* 10.6 (2012), pp. 1134–1145.

[114] Sami Mnasri et al. "A new multi-agent particle swarm algorithm based on birds accents for the 3D indoor deployment problem". In: *ISA transactions* 91 (2019), pp. 262–280.

[115] Ahmed Mahdi Jubair et al. "Social class particle swarm optimization for variable-length Wireless Sensor Network Deployment". In: *Applied Soft Computing* 113 (2021), p. 107926.

[116] Yu Qiao, Hung-Yao Hsu, and Jeng-Shyang Pan. "Behavior-based Grey Wolf Optimizer for Wireless Sensor Network Deployment Problem". In: *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)* 32.1/2 (2020).

[117] Shipeng Wang et al. "A virtual force algorithm-Lévy-embedded grey wolf optimization algorithm for wireless sensor network coverage optimization". In: *Sensors* 19.12 (2019), p. 2735.

[118] Zhendong Wang et al. "Node coverage optimization algorithm for wireless sensor networks based on improved grey wolf optimizer". In: *Journal of Algorithms & Computational Technology* 13 (2019), p. 1748302619889498.

[119] Zhaoming Miao et al. "Grey wolf optimizer with an enhanced hierarchy and its application to the wireless sensor network coverage optimization problem". In: *Applied Soft Computing* 96 (2020), p. 106602.

[120] Christian Blum et al. "Hybrid metaheuristics in combinatorial optimization: A survey". In: *Applied soft computing* 11.6 (2011), pp. 4135–4151.

[121] Christian Blum and Andrea Roli. "Hybrid metaheuristics: an introduction". In: *Hybrid Metaheuristics*. Springer, 2008, pp. 1–30.

[122] TO Ting et al. "Hybrid metaheuristic algorithms: past, present, and future". In: *Recent advances in swarm intelligence and evolutionary computation* (2015), pp. 71–83.

[123] Ying Xu et al. "Hybrid multi-objective evolutionary algorithms based on decomposition for wireless sensor network coverage optimization". In: *Applied Soft Computing* 68 (2018), pp. 268–282.

[124] Sami Mnasri et al. "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*. Springer. 2017, pp. 367–381.

[125] Jingwen Tian, Meijuan Gao, and Guangshuang Ge. "Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm". In: *EURASIP Journal on Wireless Communications and Networking* 2016.1 (2016), pp. 1–11.

[126] Sami Mnasri et al. "The 3D redeployment of nodes in Wireless Sensor Networks with real testbed prototyping". In: *International Conference on Ad-Hoc Networks and Wireless*. Springer. 2017, pp. 18–24.

[127] M Yuvaraja, R Sabitha, and S Karthik. "Hybrid PSO-Bat Algorithm with Fuzzy Logic based Routing Technique for Delay Constrained Lifetime Enhancement in Wireless Sensor Networks". In: *Journal of Internet Technology* 21.2 (2020), pp. 479–487.

[128] Donghui Ma and Qianqian Duan. "A hybrid-strategy-improved butterfly optimization algorithm applied to the node coverage problem of wireless sensor networks". In: *Mathematical Biosciences and Engineering* 19.4 (2022), pp. 3928–3952.

[129] Adda Boualem, Marwane Ayaida, and Cyril De Runz. "Hybrid model approach for wireless sensor networks coverage improvement". In: *2020 8th International Conference on Wireless Networks and Mobile Communications (WINCOM)*. IEEE. 2020, pp. 1–6.

[130] Chia-Pang Chen et al. "A hybrid memetic framework for coverage optimization in wireless sensor networks". In: *IEEE transactions on cybernetics* 45.10 (2014), pp. 2309–2322.

[131] Timóteo Holanda et al. "A hybrid algorithm for deployment of sensors with coverage and connectivity constraints". In: *Int. J. Adv. Eng. Res. Sci.* 6.3 (2019), pp. 13–19.

[132] Tripatjot Singh Panag and Jaspreet Singh Dhillon. "Maximal coverage hybrid search algorithm for deployment in wireless sensor networks". In: *Wireless Networks* 25.2 (2019), pp. 637–652.

[133] Yasser El Khamlichi et al. "A hybrid algorithm for optimal wireless sensor network deployment with the minimum number of sensor nodes". In: *Algorithms* 10.3 (2017), p. 80.

[134] Xinggang Fan et al. "Coverage hole elimination based on sensor intelligent redeployment in WSN". In: *The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent*. IEEE. 2014, pp. 336–339.

[135] Hicham Deghbouch and Fatima Debbat. "A hybrid bees algorithm with grasshopper optimization algorithm for optimal deployment of wireless sensor networks". In: *Inteligencia Artificial* 24.67 (2021), pp. 18–35.

[136] Shahrzad Saremi, Seyedali Mirjalili, and Andrew Lewis. "Grasshopper optimisation algorithm: theory and application". In: *Advances in Engineering Software* 105 (2017), pp. 30–47.

[137] Darrall Henderson, Sheldon H Jacobson, and Alan W Johnson. "The theory and practice of simulated annealing". In: *Handbook of metaheuristics*. Springer, 2003, pp. 287–319.

[138] Mohssen Mohammed, Muhammad Badruddin Khan, and Eihab Bashier Mohammed Bashier. *Machine learning: algorithms and applications*. Crc Press, 2016.

[139] Mohammad Abu Alsheikh et al. "Machine learning in wireless sensor networks: Algorithms, strategies, and applications". In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 1996–2018.

[140] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. "Deep learning in mobile and wireless networking: A survey". In: *IEEE Communications surveys & tutorials* 21.3 (2019), pp. 2224–2287.

[141] Lintao Ye, Sandip Roy, and Shreyas Sundaram. "Resilient sensor placement for Kalman filtering in networked systems: Complexity and algorithms". In: *IEEE Transactions on Control of Network Systems* 7.4 (2020), pp. 1870–1881.

[142] Vasileios Tzoumas, Ali Jadbabaie, and George J Pappas. "Sensor placement for optimal Kalman filtering: Fundamental limits, submodularity, and algorithms". In: *2016 American Control Conference (ACC)*. IEEE. 2016, pp. 191–196.

[143] Mark Wei Ming Seah et al. "Achieving coverage through distributed reinforcement learning in wireless sensor networks". In: *2007 3rd international Conference on intelligent sensors, sensor networks and information*. IEEE. 2007, pp. 425–430.

[144] Yung Po Tsang et al. "Multi-objective mapping method for 3D environmental sensor network deployment". In: *IEEE Communications Letters* 23.7 (2019), pp. 1231–1235.

[145] Faten Hajjej et al. "A distributed coverage hole recovery approach based on reinforcement learning for Wireless Sensor Networks". In: *Ad Hoc Networks* 101 (2020), p. 102082.

[146] Luis Orlando Philco, Luis Marrone, and Emily Estupiñan. "MiA-CODER: A Multi-Intelligent Agent-Enabled Reinforcement Learning for Accurate Coverage Hole Detection and Recovery in Unequal Cluster-Tree-Based QoSensing WSN". In: *Applied Sciences* 11.23 (2021), p. 11134.

[147] André Borrmann et al. *Building information modeling: Why? what? how?* Springer, 2018.

[148] NBS. *What is Building Information Modelling*. Accessed on February 29, 2024. 2016. URL: https://www.thenbs.com/knowledge/what-is-building-information-modelling-bim.

[149] Rebekka Volk, Julian Stengel, and Frank Schultmann. "Building Information Modeling (BIM) for existing buildings—Literature review and future needs". In: *Automation in construction* 38 (2014), pp. 109–127.

[150] Liu Liu et al. "Indoor navigation supported by the Industry Foundation Classes (IFC): A survey". In: *Automation in Construction* 121 (2021), p. 103436.

[151] Inno Institute. *BIM Image*. Accessed on February 28, 2024. 2024. URL: https://inno-institute.com/bim.html.

[152] International Organization for Standardization. *ISO 70303:2017(en) Nuclear energy – Categorization of radioactive sources*. Accessed: March 18, 2024. 2017. URL: https://www.iso.org/standard/70303.html.

[153] bim.tech.fr. *IfcRoot*. https://bim.tech.fr/ifc/IfcRoot.html. Accessed: March 18, 2024.

[154] Thomas Krijnen et al. "Validation and inference of geometrical relationships in IFC". In: *Proceedings of the 37th International Conference of CIB W*. Vol. 78. 2020, pp. 98–111.

[155] *IFC FacetedBrep*. BuildingSMART. URL: https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/schema/ifcgeometricmodelresource/lexical/ifcfacetedbrep.htm.

[156] Eid Emary, Hossam M Zawbaa, and Aboul Ella Hassanien. "Binary grey wolf optimization approaches for feature selection". In: *Neurocomputing* 172 (2016), pp. 371–381.

[157] Ashraf Hossain, Subit Chakrabarti, and Prabir Kumar Biswas. "Impact of sensing model on wireless sensor network coverage". In: *IET Wireless Sensor Systems* 2.3 (2012), pp. 272–281.

[158] Janice M Keenan and Andrew J Motley. "Radio coverage in buildings". In: *British telecom technology Journal* 8.1 (1990), pp. 19–24.

[159] Khaoula Zaimen et al. "Connectivity Repair Heuristics for Stationary Wireless Sensor Networks". In: *IEEE International Conference on Communications*. 2023.

[160] Mohamed Younis et al. "Topology management techniques for tolerating node failures in wireless sensor networks: A survey". In: *Computer networks* 58 (2014), pp. 254–283.

[161] Karima Bouyahia and Mahfoud Benchaiba. "CRVR: Connectivity Repairing in Wireless Sensor Networks with Void Regions". In: *Journal of Network and Systems Management* 25.3 (2017), pp. 536–557.

[162] Sookyoung Lee and Mohamed Younis. "Recovery from multiple simultaneous failures in wireless sensor networks using minimum Steiner tree". In: *Journal of Parallel and Distributed Computing* 70.5 (2010), pp. 525–536.

[163] Sookyoung Lee and Mohamed Younis. "Optimized relay node placement for connecting disjoint wireless sensor networks". In: *Computer Networks* 56.12 (2012), pp. 2788–2804.

[164] Sookyoung Lee, Mohamed Younis, and Meejeong Lee. "Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance". In: *Ad Hoc Networks* 24 (2015), pp. 1–19.

[165] Virender Ranga, Mayank Dave, and Anil Kumar Verma. "Relay node placement for lost connectivity restoration in partitioned wireless sensor networks". In: *Proceedings of International Conference on Electronics and Communication Systems (ECS 2015)*. 2015, pp. 170–175.

[166] Shangdong Liu et al. "Distributed Connectivity Restoration Algorithm with Optimal Repair Path in Wireless Sensor and Actor Networks". In: *Journal of Sensors* 2022 (2022).

[167] Hong Zeng and Zhiping Kang. "Relay node placement to restore connectivity in wireless sensor networks". In: *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*. IEEE. 2017, pp. 301–305.

[168] Wassila Lalouani, Mohamed Younis, and Nadjib Badache. "Optimized repair of a partitioned network topology". In: *Computer Networks* 128 (2017), pp. 63–77.

[169] Osama Moh'd Alia and Alaa Al-Ajouri. "Maximizing wireless sensor network coverage with minimum cost using harmony search algorithm". In: *IEEE Sensors Journal* 17.3 (2016), pp. 882–896.

[170] Ning-ning Qin and Jia-le Chen. "An area coverage algorithm for wireless sensor networks based on differential evolution". In: *International Journal of Distributed Sensor Networks* 14.8 (2018), p. 1550147718796734.

[171] Satinder Singh Mohar, Sonia Goyal, and Ranjit Kaur. "Optimized Sensor Nodes Deployment in Wireless Sensor Network Using Bat Algorithm". In: *Wireless Personal Communications* 116.4 (2021), pp. 2835–2853.

[172] Mohamed Amin Benatia et al. "Impact of radio propagation in buildings on WSN's lifetime". In: *2014 Global Summit on Computer & Information Technology (GSCIT)*. IEEE. 2014, pp. 1–6.

[173] YE Mohammed, AS Abdallah, and YA Liu. "Characterization of indoor penetration loss at ISM band". In: *Asia-Pacific Conference on Environmental Electromagnetics, 2003. CEEM 2003. Proceedings.* IEEE. 2003, pp. 25–28.

[174] Bin Xu et al. "Adaptive differential evolution with multi-population-based mutation operators for constrained optimization". In: *Soft Computing* 23 (2019), pp. 3423–3447.

[175] W Long and TB Wu. "Improved grey wolf optimization algorithm coordinating the ability of exploration and exploitation". In: *Control and decision* 32.10 (2017), pp. 1749–1757.

[176] Shanu Verma, Millie Pant, and Vaclav Snasel. "A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems". In: *Ieee Access* 9 (2021), pp. 57757–57791.

[177] Eckart Zitzler and Lothar Thiele. "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach". In: *IEEE transactions on Evolutionary Computation* 3.4 (1999), pp. 257–271.

[178] Manuel López-Ibáñez et al. "The irace package: Iterated racing for automatic algorithm configuration". In: *Operations Research Perspectives* 3 (2016), pp. 43–58.

[179] Paul G Huray. *Maxwell's equations*. John Wiley & Sons, 2009.

[180] Hind Taud and JF Mas. "Multilayer perceptron (MLP)". In: *Geomatic approaches for modeling land change scenarios* (2018), pp. 451–455.

[181] Andrej Krenker, Janez Bešter, and Andrej Kos. "Introduction to the artificial neural networks". In: *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech* (2011), pp. 1–18.

[182] Leire Azpilicueta et al. “A ray launching-neural network approach for radio wave propagation analysis in complex indoor environments”. In: *IEEE Transactions on Antennas and Propagation* 62.5 (2014), pp. 2777–2786.

[183] Jinxiao Wen et al. “Path loss prediction based on machine learning methods for aircraft cabin environments”. In: *Ieee Access* 7 (2019), pp. 159251–159261.

[184] M Ayadi, A Ben Zineb, and S Tabbane. “A UHF path loss model using learning machine for heterogeneous networks”. In: *IEEE Transactions on Antennas and Propagation* 65.7 (2017), pp. 3675–3683.

[185] Joseph Isabona and Viranjay M Srivastava. “Hybrid neural network approach for predicting signal propagation loss in urban microcells”. In: *2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*. IEEE. 2016, pp. 1–5.

[186] Aymen Ben Zineb and Mohamed Ayadi. “A multi-wall and multi-frequency indoor path loss prediction model using artificial neural networks”. In: *Arabian Journal for Science and Engineering* 41.3 (2016), pp. 987–996.

[187] Erik Ostlin, Hans-Jürgen Zepernick, and Hajime Suzuki. “Macrocell path-loss prediction using artificial neural networks”. In: *IEEE Transactions on Vehicular Technology* 59.6 (2010), pp. 2735–2747.

[188] Ladislav Polak et al. “Received signal strength fingerprinting-based indoor location estimation employing machine learning”. In: *Sensors* 21.13 (2021), p. 4605.

[189] O Perrault, J-P Rossi, and Thierry Balandier. “Predicting field strength with a neural ray-tracing model”. In: *Proceedings of GLOBECOM'96. 1996 IEEE Global Telecommunications Conference*. Vol. 2. IEEE. 1996, pp. 1167–1171.

[190] Keiron O'Shea and Ryan Nash. “An introduction to convolutional neural networks”. In: *arXiv preprint arXiv:1511.08458* (2015).

[191] Kehai Qiu et al. “Pseudo Ray-Tracing: Deep Leaning Assisted Outdoor mm-Wave Path Loss Prediction”. In: *IEEE Wireless Communications Letters* 11.8 (2022), pp. 1699–1702.

[192] Sotirios P Sotiroudis, Sotirios K Goudos, and Katherine Siakavara. “Deep learning for radio propagation: Using image-driven regression to estimate path loss in urban areas”. In: *ICT Express* 6.3 (2020), pp. 160–165.

[193] Siyi Huang and Xingqi Zhang. “CNN-based received signal strength prediction: A frequency conversion scheme”. In: *Electronics Letters* 59.7 (2023), e12766.

[194] Aristeidis Seretis and Costas D Sarris. "Toward physics-based generalizable convolutional neural network models for indoor propagation". In: *IEEE Transactions on Antennas and Propagation* 70.6 (2022), pp. 4112–4126.

[195] T Imai, K Kitao, and M Inomata. "Radio propagation prediction model using convolutional neural networks by deep learning". In: *2019 13th European Conference on Antennas and Propagation (EuCAP)*. IEEE. 2019, pp. 1–5.

[196] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[197] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer. 2015, pp. 234–241.

[198] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[199] Vishnu V Ratnam et al. "FadeNet: Deep learning-based mm-wave large-scale channel fading prediction and its applications". In: *IEEE Access* 9 (2020), pp. 3278–3290.

[200] Xin Zhang et al. "Cellular network radio propagation modeling with deep convolutional neural networks". In: *Proceedings of the 26th ACM SIGKDD International Conference on knowledge discovery & data mining*. 2020, pp. 2378–2386.

[201] Stefanos Bakirtzis et al. "EM DeepRay: an expedient, generalizable, and realistic data-driven indoor propagation model". In: *IEEE Transactions on Antennas and Propagation* 70.6 (2022), pp. 4140–4154.

[202] Shuhang Wang et al. "U-net using stacked dilated convolutions for medical image segmentation". In: *arXiv preprint arXiv:2004.03466* (2020).

[203] Getao Du et al. "Medical Image Segmentation based on U-Net: A Review." In: *Journal of Imaging Science & Technology* 64.2 (2020).

[204] Wei Yao et al. "Pixel-wise regression using U-Net and its application on pansharpening". In: *Neurocomputing* 312 (2018), pp. 364–371.