REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Ecole Nationale Polytechnique

Département d'Electronique

Mémoire de projet de fin d'études
pour l'obtention du diplôme d'ingénieur d'état en électronique

---

Déploiement d'un réseau de capteurs sans fil dans un bâtiment intelligent en utilisant les algorithmes génétiques et les forêts aléatoires

---

ARFI Nadir

MEHENNI Walid

Présenté et soutenu publiquement le (08/07/2021)

Composition du jury :

M. Cherif LARBES	**Professeur**	**ENP**	**Président**
Mme. Nour El-Houda BENALIA	**Docteur**	**ENP**	**Promotrice**
Mlle. Soumaya FERHAT TALEB	**Doctorante**	**ENP**	**Co-Promotrice**
Mme. Nesrine BOUADJENEK	**Docteur**	**ENP**	**Examinatrice**

ENP 2021

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Ecole Nationale Polytechnique

Département d'Electronique

Mémoire de projet de fin d'études
pour l'obtention du diplôme d'ingénieur d'état en électronique

---

Déploiement d'un réseau de capteurs sans fil dans un bâtiment intelligent en utilisant les algorithmes génétiques et les forêts aléatoires

---

ARFI Nadir

MEHENNI Walid

Présenté et soutenu publiquement le (08/07/2021)

Composition du jury :

M. Cherif LARBES	**Professeur**	**ENP**	**Président**
Mme. Nour El-Houda BENALIA	**Docteur**	**ENP**	**Promotrice**
Mlle. Soumaya FERHAT TALEB	**Doctorante**	**ENP**	**Co-Promotrice**
Mme. Nesrine BOUADJENEK	**Docteur**	**ENP**	**Examinatrice**

ENP 2021

ملخص

في مشروع نهاية الدراسة هذا ، درسنا مشكلة نشر شبكات الاستشعار اللاسلكية في مبنى ذكي. وكان الهدف هو تطوير وسيلة لإنشاء مخطط نشر فعال ، وضمان تقديم حلول جيدة النوعية . بدأنا بتقديم المباني ذكية و شبكات الاستشعار اللاسلكية. ثم قمنا بنمذجة المعلمات الرئيسية لنشر شبكات الاستشعار اللاسلكية ، وهي التكلفة والاتصال والتغطية والتغطية الزائدة. بعد ذلك ، قدمنا وطبقنا خوارزميات وراثية متعددة الأهداف كحل لمشكلة النشر. بعد اختبار هذه الخوارزميات. أظهرت النتائج التي تم الحصول عليها أن هذه الخوارزميات فعالة عندما يتعلق الأمر بمبنى صغير. من ناحية أخرى ، بمجرد زيادة حجم المبنى ، يتدهور أداء الخوارزميات المدروسة. للتغلب على هذه المشكلة قدمنا مفهوم التهجين لتحسين الأداء. يتم ذلك عن طريق الجمع بين الخوارزميات الجينية متعددة الأهداف المستخدمة سابقًا مع خوارزميات شجرة القرار. أظهرت النتائج التي تم الحصول عليها أداءً جيدًا جدًا في الحالتين ذات الأحجام المختلفة.

الكلمات الرئيسية: شبكة الاستشعار اللاسلكية ، النشر ، الاتصال ، التغطية ، التحسين متعدد الأهداف ، الخوارزمية التطورية. خوارزميات شجرة القرار, خوارزميات التهجين .

Abstract

In this graduation project, we studied the problem of deploying Wireless Sensor Networks within an intelligent building. The main goal was to develop a solution generate a deployment plan of a WSN while ensuring good quality solutions. We started this project with a presentation of smart buildings and the concepts of WSNs. Then, we proposed and implemented multi-objective genetic algorithms as a solution for the deployment problem. The results obtained showed that the performance of the studied algorithms degrades when it comes to large buildings. To overcome this problem we to hybridization in order to improve the performances. This was done by combining the previously used MOGA with Random forest algorithm. The results obtained showed a very good performance on the two instances of different sizes.

Keywords : Wireless sensor network, deployment, multi-objective optimization, Metaheuristics, Machine learning, hybridization, Random forest.

Résumé

Dans ce projet de fin d'études, nous avons étudié le problème du déploiement de réseaux de capteurs sans fil dans un bâtiment intelligent. L'objectif principal était de développer une solution permettant de générer un plan de déploiement d'un RCSF tout en assurant une bonne qualité des solutions. Nous avons commencé ce projet par une présentation des bâtiments intelligents et des concepts de RCSFs. Ensuite, nous avons proposé et implémenté des algorithmes génétiques multi-objectifs comme solution au problème de déploiement. Les résultats obtenus ont montré que les performances des algorithmes étudiés se dégradent lorsqu'il s'agit de grands bâtiments. Pour surmonter ce problème, nous avons eu recours à l'hybridation afin d'améliorer les performances. Ceci a été fait en combinant le MOGA précédemment utilisé avec l'algorithme Random forest. Les résultats obtenus ont montré une très bonne performance sur les deux instances de tailles différentes.

Mots clés : Réseau de capteurs sans fil, déploiement, optimisation multi-objectif, Métaheuristiques, Hybridation, Machine learning, Random forest.

Dédicaces

Je dédie ce travail,

À mon père, mon modèle et ma source d'inspiration, tu m'as toujours poussé et aidé à devenir la meilleure version de moi-même. Tu as toujours été à mes côtés pour me soutenir et m'encourager. Tes conseils ont toujours guidé mes pas vers la réussite. Je te dois ce que je suis aujourd'hui et ce que je serai demain et je ferai toujours de mon mieux pour rester ta fierté.

À ma très chère maman, la femme qui m'a amené au monde, celle qui a tout sacrifié pour moi. Je suis le fruit de ton amour, ta bienveillance me guide et ta présence à mes côtés a toujours été ma source de bonheur.

À la mémoire de mes grands-parents que Dieu les accueille dans son vaste paradis.

À mes deux frères aînés Redouane et Redha, À mes belles-soeurs et mes deux adorables nièces Lyna et Anya.

À mes amis les plus proches depuis le lycée Mahieddine, Salim, Amine, Samy, Hamza, Khadidja, Lyna, Lina.

À mes amis extraordinaires que j'ai rencontrés à l'école Ramzi, Walid, Dhayaedine, Djamel, Zaki, Rostane, Ilyas, Issem, Arysse, Cerine, Feriel et Meriem.

À tous mes camarades de classe, qui ont fait de ces 5 années une expérience très spéciale pour moi.

Nadir ARFI

Dédicaces

Je dédie ce travail,

À ma très chère maman qui a tout donné et sacrifié pour moi, à mon frère Zakaria et ma sœur Imene qui ont toujours été là pour m'aider et soutenir pendant tout mon parcours,

À la mémoire de mon père que Dieu l'accueille dans son vaste paradis.

À mes amis d'enfance les plus proches Amine, Islam, Youcef et mon cousin Raouf.

À mes chers amis que j'ai rencontrés pendant mon parcours qui ont été plus que des camarades Issem, Meriem, Cerine, Feriel, Nadir, Ilyas, Larbi, Abderrahmane, Djamel et Dhayaedine.

À tous mes amis et à tous ceux que j'aime.

Walid MEHENNI

Remerciements

Nos profonds remerciements et sincères sentiments de gratitude à **Dr. Nour el houda BENALIA** et **Mlle. Soumaya FERHAT TALAB** nos deux promotrices, pour tout ce qu'elles ont fait pour nous, pour tout leur aide, soutien, engagement et surtout l'énergie consacrée à la direction de ce travail.

Nous remercions également l'ensemble des membres du jury de m'avoir fait l'honneur d'examiner ce travail, tout en leur adressant mon plus profond respect.

Nous souhaitons par la même occasion remercier tous nos enseignants du département d'électronique de **l'École Nationale Polytechnique**. Enfin, nous remercions tous les acteurs de l'école qui ont contribué à nous offrir tous les outils nécessaires à la réussite de notre formation d'ingénieur.

Pour finir, nous remercions infiniment nos familles ainsi que tous nos proches pour le soutien tout au long de ce projet.

Table des matières

Liste des tableaux

Table des figures

Liste des abréviations

2D : 2 dimensions

3D : 3 dimensions

AG : Algorithme génétique

AGMO : Algorithme génétique multi-objectif

IoT : Internet of Things

NP : Non-deterministic Polynomial time

NSGA : Non dominated sorting genetic algorithm

QoS : Quality of Service

RSSI : Recieved Signal Streght Indication

SPEA : Strength Pareto evolutionary algorithm

RCSF : Réseau de capteurs sans fil

Rd : Rayon de détection

DT : Arbre de décision (Decision tree)

RF : Forêt aléatoire (Random forest)

RN : Réseaux de neurones

RMSE : Ecart quadratique moyen

Introduction générale

Contexte et motivation Les premiers bâtiments jamais construits étaient des abris faits de pierres, de bâtons, de peaux d'animaux et d'autres matériaux naturels. S'ils ne ressemblaient guère à l'acier et au verre qui composent les villes modernes, ces premières structures avaient le même objectif, à savoir offrir un espace confortable aux personnes qui s'y trouvaient. Une sorte d'intelligence est implémentée au sein d'un bâtiment par l'utilisation de réseaux de capteurs sans fil (RCSF), cela est maintenant possible grâce à l'amélioration de la technologie moderne et la miniaturisation des dispositifs électroniques. Un bâtiment intelligent est capable d'interagir et de prendre des décisions en analysant les données acquises en temps réel par le RCSF. Ces systèmes collectent des informations liées à l'environnement (température, humidité, luminosité, présence...) et prennent des décisions pour optimiser la performance opérationnelle sur tous les aspects. En effet, pour deployer un RCSF, un certain nombre de contraintes doit être respecté pour avoir un réseau performant et répondant aux exigences de l'application à laquelle il est destiné. Généralement, ces contraintes sont principalement liées à la connectivité de tous les capteurs, à la couverture de la zone considérée, à leur durée de vie, etc.

Problématique La problématique de notre projet s'inscrit dans ce contexte. Elle consiste à étudier la démarche à suivre pour la conception et le déploiement de ce type de réseau. Le déploiement d'un RCSF comprend la détermination du nombre et de la disposition des nœuds. Ces nœuds doivent former un réseau qui répond aux taux de connectivité, de couverture et de sur-couverture souhaités à moindre coût. Au cours des dernières décennies, des méthodes d'optimisation et des méta-heuristiques ont été appliquées pour le design et le déploiement des RCSF. Les solutions proposées ne traitent qu'une ou deux contraintes au maximum. Cependant, le développement d'un outil en mesure de considérer trois objectifs ou plus est une nécessité primordiale pour la conception d'un RCSF efficace. Pour y parvenir, une bonne modélisation du problème doit être élaborée. En effet, les différents

critères de déploiement que nous avons choisis sont : le coût, qui est proportionnel au nombre de capteurs déployés, la couverture du bâtiment qui doit être maximisée afin d'avoir une bonne portée, la connectivité du réseau, qui doit également être optimisée afin d'assurer une meilleure qualité de service, et enfin, la sur-couverture visant à éviter de déployer des capteurs de manière redondante et sans utilité. En plus de cela, une méthode d'optimisation multi-objectif doit être appliquée pour déterminer le nombre et l'emplacement des différents nœuds qui composent le réseau.

Objectif L'objectif principal de notre projet est d'améliorer les performances d'une méthode d'optimisation multi-objectifs en s'appuyant sur des techniques d'hybridation pour le cas des grands bâtiments. Il s'agit de combiner plusieurs méthodes afin de tirer les avantages et améliorer les performances d'un algorithme d'optimisation. Ce domaine a connu un très grand succès dans la communauté scientifique ce qui a suscité notre intérêt pour ce genre d'approches.

Organisation du mémoire Ce mémoire est organisé en trois chapitres. Dans le premier chapitre, nous aurons un aperçu général des bâtiments intelligents et de la manière dont nous pouvons y intégrer l'intelligence en nous appuyant sur des techniques d'optimisation qui assurent un déploiement efficace d'un RCSF. Ensuite, dans le deuxième chapitre, nous proposerons l'algorithme génétique comme première solution au problème du déploiement et nous nous concentrerons sur son adaptation à différentes tailles de bâtiments. Nous effectuerons plusieurs simulations pour déterminer les performances de notre solution en termes de qualité et de temps de calcul. Enfin, dans le dernier chapitre, nous proposerons une technique d'hybridation à mettre en œuvre en parallèle avec le AGMO afin de réduire son temps d'exécution et de maintenir une solution de bonne qualité. De la même manière que dans le chapitre précédent, nous effectuons plusieurs simulations afin de comparer les performances et de valider notre solution.

Chapitre 1

Critères et stratégies de déploiement d'un RCSF dans un bâtiment intelligent

1.1 Introduction

La domotique [1], parfois appelée smart home, ou maison intelligente, est devenue un terme de plus en plus populaire. Cependant, pour de nombreuses personnes, le terme est encore obscur. En effet, la domotique est un concept technologique visant à mettre en place un ensemble de techniques modernes pour implémenter une sorte d'intelligence au sein d'une maison ou un bâtiment. Depuis de nombreuses années, la domotique est considérée comme un domaine très prometteur pour le développement des technologies électroniques qui vise à apporter des solutions techniques pour répondre aux besoins de confort (gestion de l'énergie, optimisation de l'éclairage et du chauffage), de sécurité (alarmes) et de communication (télécommande, signaux visuels ou sonores, etc.) dans les habitations, lieux publics, etc.

L'omniprésence croissante de capteurs à faible coût et de technologies sans fil modifie radicalement les environnements intérieurs dans lesquels nous travaillons et vivons. De ce fait, le bâtiment intelligent [2] s'engage dans la continuité de cette démarche et cela permet d'obtenir des bâtiments complexes qui correspondent parfaitement aux besoins de la vie quotidienne.

Au début de ce chapitre, nous aurons un aperçu général sur les bâtiments intelligents et la façon dont ils peuvent surmonter de nombreux problèmes grâce à l'utilisation des RCSFs. Par la suite, nous présentons ces derniers, leurs architectures, leurs applications et les défis qu'ils posent lors du déploiement. Nous définirons également quelques critères nécessaires pour avoir un déploiement optimisé qui puisse garantir une utilisation efficace du RCSF. Enfin, nous abordons quelques stratégies qui ont été utilisées pour la résolution du problème de déploiement d'un RCSF.

1.2 Bâtiment intelligent

1.2.1 Qu'est ce qu'un bâtiment intelligent

Le bâtiment intelligent [3] est une structure complexe qui utilise des dispositifs de l'IoT (capteurs, logiciels) pour surveiller diverses caractéristiques du bâtiment et protéger ainsi la santé et la sécurité des occupants. L'objectif principal de ces technologies est de le rendre plus performant sur le plan opérationnel afin d'améliorer la productivité des employés et de réduire son impact sur l'environnement.

1.2.2 Avantages majeurs des bâtiments intelligents

Grâce à l'utilisation des capteurs, tels que les détecteurs de mouvement, des données exploitables sur la façon dont les bâtiments sont utilisés peuvent être collectées pour lui permettre d'être plus performant. Voici cinq avantages clés [4] des bâtiments intelligents :

1.2.2.1 Réduction de la consommation d'énergie

La consommation d'énergie d'un bâtiment peut être réduite d'environ 5 à 35% [4] grâce à l'utilisation de technologies intelligentes. Cela se traduit par de considérables économies financières, ainsi que par une approche beaucoup plus efficace et efficiente pour atteindre les objectifs écologiques.

1.2.2.2 Amélioration de l'efficacité du bâtiment

Les capteurs fournissent des données sur la façon dont le bâtiment est utilisé. Cela permet aux systèmes intelligents de faire les ajustements nécessaires sur l'utilisation des infrastructures tel que la climatisation. Les capteurs permettent également d'identifier les zones sur-utilisées et sous-utilisées du bâtiment, ce qui permet d'optimiser l'utilisation de l'espace.

1.2.2.3 Maintenance prédictive

Les coûts de maintenance peuvent être importants lorsqu'ils sont gérés manuellement. Cependant, sans maintenance, les équipements des bâtiments nécessitent des remplacements beaucoup plus fréquents. Cependant, les bâtiments intelligents permettent une maintenance prédictive plus simple. Des capteurs peuvent détecter les performances du bâtiment et activer les procédures de maintenance avant qu'une alerte ne soit déclenchée.

1.2.2.4 Augmentation de la productivité

Les bâtiments intelligents ont été spécialement conçus pour offrir une expérience plus confortable à leurs occupants. Ils peuvent améliorer les normes et garantir que les considérations de santé et de sécurité sont respectées, tout en veillant à ce qu'elles soient mises en œuvre de manière rentable. Les bâtiments intelligents rendent les gens plus productifs en surveillant continuellement l'utilisation du bâtiment et en ajustant les systèmes pour garantir que les occupants disposent des installations dont ils ont besoin.

1.2.2.5 Utilisation meilleure des ressources

Les données générées par un bâtiment intelligent fournissent des informations clés qui peuvent être intégrées dans la planification et rendre l'utilisation des ressources plus efficace. Il n'est plus nécessaire de s'appuyer sur des suppositions, car elles peuvent être alimentées par des renseignements authentiques en temps réel.

1.3 Réseau de capteurs sans fil

1.3.1 Qu'est ce qu'un RCSF ?

Le RCSF est un réseau sans fil composé d'un grand nombre de petits dispositifs [5], appelés nœuds capteurs. Ces derniers sont répartis dans l'espace avec une ou plusieurs stations de base chargées de la collecte de données. Un RCSF est souvent caractérisé par sa capacité à surveiller périodiquement des phénomènes physiques grâce à l'acquisition et le traitement de données par le biais d'une entité principale, également appelée station de base.

1.3.2 Architecture d'un RCSF

Un RCSF peut être connectée à une infrastructure ou à l'internet via une passerelle. Cette transformation de données issues d'un RCSF bénéficiera des techniques développées pour les technologies de l'informatique en nuage tel que le Cloud Computing [6]. Cela permet aux utilisateurs d'accéder aux données collectées à distance et de surveiller leurs systèmes en temps réel. D'ailleurs, un RCSF est constitué de trois éléments [7] :

1.3.2.1 Noeud capteur

Le nœud capteur est un dispositif multifonctionnel qui se compose d'une unité de détection, d'une unité de traitement, d'une unité de stockage, d'une unité de communication, d'une unité d'alimentation, d'une unité de mobilisation, etc. Les nœuds de capteurs ont la capacité de détecter la cible ou le phénomène qui se produit dans leur champ de détection, de traiter les données sensorielles et de les transmettre à la station de base.

1.3.2.2 Noeud routeur

Appelé aussi nœud puit ou sink, ce dispositif est également un capteur qui peut recevoir, traiter et enregistrer des données provenant des nœuds capteurs, donc il détient des capacités supérieures en termes de puissances de traitement, de mémoire et d'énergie.

1.3.2.3 Station de base

La station de base est un point central qui sert à contrôler le réseau pour extraire des données puis les envoyer aux différents nœuds du réseau.

1.3.3 Limites des RCSF

Mais malgré ses nombreux avantages, la technologie RCSF souffre de certains inconvénients communs tels que ceux qu'on trouve dans les réseaux industriels filaires :

- Tous les nœuds du RCSF doivent utiliser le même protocole de communication et de plus, les différentes versions doivent être compatibles. Par exemple, un nœud qui utilise le Wi-Fi ne peut pas communiquer avec un nœud qui prend uniquement en charge le protocole du Bluetooth.

- La communication RCSF peut être influencée par les conditions météorologiques et par les obstacles naturels tels que les arbres, etc.

- Les capteurs doivent être disposés judicieusement de façon telle que la technologie de communication choisie puisse assurer la protection contre les interférences.

1.3.4 Applications des RCSFs

1.3.4.1 Découvertes de catastrophes naturelles

On peut créer un réseau autonome en dispersant des nœuds capteurs dans la nature qui peuvent signaler des événements [8] tels que feux de forêts, tempêtes ou inondations, permettant une intervention beaucoup plus rapide et efficace des secours.

1.3.4.2 Domotique

Dans le domaine de la domotique [1], les sociétés de gestion des installations doivent être capables de gérer des bâtiments intelligents dans un certain délai en raison des contrats de location. Pour cela, l'automatisation des bâtiments et les capteurs nécessaires doivent être installés de manière économique et d'une façon différente de la logique câblée.

1.3.4.3 Contrôle de la pollution

On pourrait disperser des capteurs au-dessus d'un emplacement industriel pour détecter et contrôler des fuites de gaz ou de produits chimiques. Ces applications permettent de donner l'alerte en un temps record et de pouvoir suivre son évolution [9] .

1.3.4.4 Agriculture

Des nœuds capteurs peuvent être incorporés dans la terre. On peut ensuite avoir des informations sur l'état du champ [10] (déterminer par exemple les secteurs les plus secs afin de les arroser en priorité).

1.3.4.5 Surveillance médicale

En implantant sous la peau de mini capteurs vidéo, on peut recevoir des images en temps réel d'une partie du corps sans aucune chirurgie pendant environ 24h. On peut ainsi surveiller la progression d'une maladie ou la reconstruction d'un muscle [11].

1.3.4.6 Militaire

Les RCSFs ont été initialement développés pour des applications de surveillance des champs de batailles [12], la surveillance des munitions et des équipements, reconnaissance des forces ennemies.

1.3.5 Défis du déploiement d'un RCSF

1.3.5.1 Consommation d'énergie

Ce facteur est très important à prendre en considération lors du déploiement d'un RCSF, il faut donc s'assurer que la consommation d'énergie soit minimale en optimisant la communication entre les nœuds du réseau par exemple en planifiant des intervalles de sommeil en cas d'inactivité [13]. On peut aussi réduire la taille des messages échangés ou même sélectionner la meilleure méthode de routage.

1.3.5.2 Puissance du signal

La puissance du signal est une mesure de la puissance présente dans un signal radio reçu. Ce facteur dépend clairement de la distance entre ces deux nœuds et il permet d'avoir une mesure sur la qualité d'une liaison entre ces deux derniers. L'indication de la puissance du

signal reçu (RSSI) décrit la force d'un signal et dépend clairement de la distance, puissance en mode réception.

1.3.5.3 Tolérance de fautes et latence

En effet, certains capteurs peuvent générer des erreurs et ne fonctionnent plus correctement à cause d'un manque d'énergie, un problème physique ou une interférence. La tolérance de fautes est la capacité de maintenir toutes les fonctionnalités du réseau sans interruption due à une erreur intervenue sur un ou plusieurs capteurs et s'assurer que ces problèmes n'affectent pas le reste du réseau. Cependant, la latence est une mesure du retard. Elle mesure le temps qu'il faut pour une donnée pour arriver à la destination sur le réseau.

1.4 Critères de déploiement

Les RCSF sont constitués d'un grand nombre de petits nœuds de capteurs déployés de manière aléatoire ou déterministe pour surveiller le champ d'intérêt. Le déploiement d'un RCSF dans un bâtiment intelligent est un problème d'optimisation qui nécessite une bonne modélisation des paramètres qui ont une influence directe sur la qualité du réseau lors du déploiement [14]. Pour cela nous allons présenter dans cette partie les différentes modélisations de ces paramètres que se soit l'espace de déploiement ou d'autres critères : coût, couverture, connectivité, durée de vie afin de choisir la modélisation adéquate de ces critères pour notre cas d'étude.

1.4.1 Espace de déploiement

Dans le déploiement d'un RCSF au sein d'un bâtiment, les capteurs et les routeurs peuvent avoir plusieurs emplacements différents et l'espace de déploiement peut être représenté en 2D et même en 3D. De ce fait, nous allons travailler avec un espace de déploiement [15] 2D ou les capteurs / routeurs seront placés sur le plafond par exemple. Cet espace est modélisé par une grille où chaque cellule de la grille représente $1m^2$ de la surface, permettant ainsi d'avoir un emplacement potentiel d'un nœud du réseau. D'après cette modélisation de l'espace nous pouvons définir la matrice de déploiement comme suit :

$$D(i,j) = \begin{cases} 1 & \text{si un capteur ou un relais est déployé dans la cellule ;} \\ 0 & \text{sinon.} \end{cases} \tag{1.1}$$

1.4.2 Coût de déploiement

L'un des objectifs principaux lors du déploiement d'un RCSF d'un point de vue économique est de réduire le plus possible le coût du déploiement sans affecter les performances et la **QoS** du réseau. Pour cela il est important de faire un bonne modélisation de ce coût pour pouvoir l'intégrer dans la fonction d'objectifs [14]. Le coût de déploiement d'un RCSF comprend le prix des routeurs et capteurs ainsi que le coût de l'installation (prix de main d'œuvre). Soit un RCSF qui contient les ensembles R= $r_1, r_2, ...r_n$ de noeuds routeurs et S=$s_1, s_2, ...s_m$ de noeuds capteurs, le coût de déploiement de ce réseau est donc égale a :

$$Coût = C_s \cdot \|S\| + C_r \cdot \|R\|. \tag{1.2}$$

C_s et C_r : sont les coûts de déploiement de d'un nœud capteur et d'un nœud routeur respectivement.
$\|S\|$ et $\|R\|$: le nombre de nœuds capteurs et de nœuds routeurs respectivement.
En utilisant la modélisation de l'espace de déploiement définit précédemment le coût peut être modélisé avec :

$$Coût = \sum_{i=0}^{l-1} \sum_{j=0}^{L-1} D(i,j) \tag{1.3}$$

Cette opération va nous donner un coût unitaire que nous allons multiplier par le coût d'un nœud capteur ensuit celui d'un nœud routeur pour avoir le coût de déploiement totale.

1.4.3 Détection des événements

D'après la littérature, deux faits concernant le dispositif de détection peuvent être observés. Le premier indique que la capacité de détection diminue à mesure que la distance augmente. Deuxièmement, la capacité de détection s'améliore lorsque le bruit diminue. On trouve principalement dans [16] deux types de modèles de détection basés sur la probabilité de détection : (i) le modèle de détection à disque binaire (ii) le modèle de détection probabiliste, les deux modèles sont montrés dans la Figure 1.1.

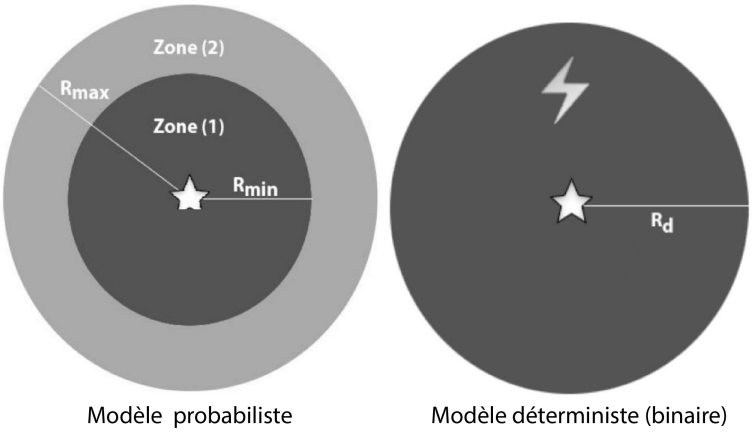

FIGURE 1.1: Les modèles de détection binaire et probabiliste

1.4.3.1 Modèle de détection déterministe (binaire)

La capacité de détection est constante dans ce modèle car la portée de détection est considérée comme un disque circulaire de rayon Rs appelé portée de détection. Ce modèle est le plus utilisé dans la littérature car il a l'avantage d'être plus simple à implémenter et il a un faible coût de calcul. Selon ce modèle, on dit qu'un événement est détecté par un nœud s'il se trouve dans son rayon de détection (R_s). Considérons (x_i, y_i) et (x_p, y_p) l'emplacement du nœud de détection si et un point quelconque p dans la zone d'intérêt et d(p, s_i) la distance euclidienne entre p et s_i. Dans le modèle de détection à disque binaire, on dit que le point p est couvert par si si l'équation suivante est vérifiée :

$$C(p, s_i) = \begin{cases} 1 & \text{si d(p,}s_i\text{)} \leq R_S\,; \\ 0 & \text{sinon.} \end{cases} \tag{1.4}$$

1.4.3.2 Modèle de détection probabiliste

En réalité, la détection d'un capteur diminue avec l'accroissement de la distance. On parle de modèles asymptomatiques. En effet, lorsqu'un événement se produit près du capteur, il a une grande probabilité d'être détecté. Cette probabilité diminue en s'éloignant du capteur et on distingue deux fonctions pour le calcul de la probabilité de détection : exponentielle, polynomiale.

Modèle exponentiel La variation de probabilité de détection est représentée par la fonction exponentielle, permettant de décrire l'effet de la distance et la qualité de détection

du capteur. Ce modèle est donnée par la fonction décrite dans [16] :

$$P_s^p = \exp(-\alpha d(s,p)) \tag{1.5}$$

Modèle polynomial Dans ce modèle, la probabilité de détection d'un événement se dégrade de manière polynomiale d'ordre (-k) et est représenté par la formule suivante :

$$P_s^p = \lambda d(s,p)^{-k} \tag{1.6}$$

où k représente la qualité de détection du capteur et λ est lié aux configurations matérielles.

1.4.4 Couverture

La couverture est l'une des mesures de la qualité de service des RCSFs. La connectivité est aussi un paramètre essentiel qu'il faut absolument prendre en considération lors du déploiement d'un tel réseau. On dit qu'une zone est couverte ou surveillée par un nœud de capteur si cette zone se trouve dans la portée de détection d'un ou plusieurs nœuds de capteur actifs. La couverture peut être classée comme suit [17] :

1.4.4.1 Couverture de champ

Ce type permet d'avoir une couverture sur une zone en fonction des exigences de l'application. Une couverture complète est essentielle dans les applications qui exigent un haut degré de précision et elle est obtenue quand le taux de couverture est égale à 1. En fait, certaines applications ne nécessitent pas une couverture complète et une couverture partielle est acceptable, ce qui est un moyen très efficace pour économiser l'énergie des capteurs et de prolonger la durée de vie du réseau puisque le nombre de capteurs déployés dans le zone est inférieur à celui requis pour une couverture complète, qui est ilustré sur la Figure 1.2.

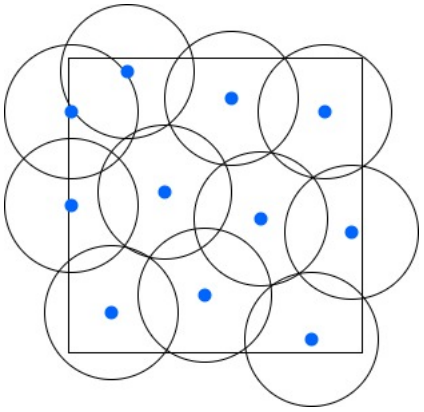

FIGURE 1.2: Modèle de couverture de champ

1.4.4.2 Couverture de cible

Dans de nombreuses applications, il suffit de surveiller quelques points d'intérêts spécifiques à la zone d'application et il n'est pas intéressant de recouvrir toute la zone. Dans ce cas, on a recours à une couverture de cible. En fait, le nombre de cibles est généralement fixe et il est donc clair que le coût de déploiement du réseau se voit réduit puisqu'il s'agit d'une couverture partielle, cela est montré sur la Figure 1.3.

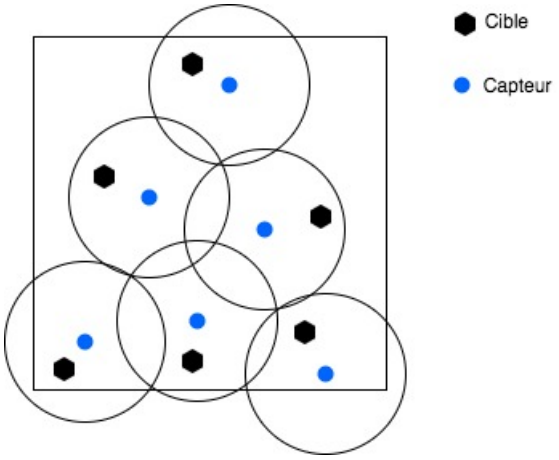

FIGURE 1.3: Modèle de couverture des points d'intérêts (cibles)

1.4.4.3 Couverture de barrière

Dans la couverture par barrière, les nœuds sont déployés de manière à former une barrière sur un chemin spécifique et à transmettre l'information s'ils détectent des activités possibles d'un intrus pour traverser la barrière. L'objectif principal est que chaque point de la zone d'intérêt soit dans le champ de couverture d'au moins un capteur.

1.4.4.4 Couverture par balayage

En fait, il suffit de couvrir périodiquement certains points d'intérêt de sorte que la couverture varie dans le temps. Par conséquent, nous pouvons surveiller un grand nombre de points d'intérêt avec un nombre minimal de nœuds de capteurs mobiles.

1.4.5 Sur-couverture

Afin de minimiser le coût de déploiement des réseaux de capteurs sans fil et de réduire les interférences entre capteurs, il est nécessaire d'éviter l'apparition de surfaces dites sur-couvertes [18], c'est-à-dire deux ou plusieurs zones couvertes. Pour cela, on utilisera l'une des représentations de la sur-couverture, qui se définit comme suit :

$$Sur - couv(i,j) = \begin{cases} 1 & \text{si la cellule (i,j) est couverte par au moins deux capteurs ;} \\ 0 & \text{sinon.} \end{cases} \tag{1.7}$$

Nous implémentons par la suite une fonction qui calcule le taux de sur-couverture à l'aide de la formule suivante :

$$Sur - couv_{RI} = \frac{\sum_{i=0}^{l-1} \sum_{j=0}^{L-1} Sur - couv(i,j)}{L \cdot l} \cdot 100 \tag{1.8}$$

1.4.6 Connectivité

Une couverture sans connectivité réduira les performances du WSN car les données n'atteindront pas les stations de base. Un réseau dans une zone définie est dit connecté si chaque nœud de capteur déployé a au moins un nœud voisin connecté [7]. Le rôle principal de la connectivité est la transmission des données vers la station de base. unique.

Les données recueillies ou détectées par les nœuds capteurs doivent être transmises à la station de base par un seul ou plusieurs sauts. Si un nœud de capteur communique directement avec la station de base pour la transmission des données, on parle alors de communication à saut. Dans la communication multi-sauts, les données détectées sont transmises à la station de base via d'autres nœuds de capteurs intermédiaires routeurs. En fait, la connectivité d'un réseau est liée à la propagation d'ondes radios. Un nœud A est dit connecté à un autre nœud B si le RSSI du signal émis par le nœud B est supérieur à la sensibilité de l'antenne

de réception du nœud A. Il existe plusieurs modèles de propagation radio qui peuvent exprimer le comportement de l'onde transmise par un nœud dans un environnement précis. Ces modèles sont présentés dans les parties suivantes :

1.4.6.1 Modèle de FRIS

Ce modèle est l'un des premiers modèles utilisés[19], il est exprime un rapport de puissance du signal reçu et celui émis par deux antennes isotropes séparées d'une distance R. Le rapport est donnée par l'équation suivante :

$$\frac{P_r}{P_t} = n * \left[\left(\frac{\lambda}{4\pi R} \right)^2 * G_t * G_r \right] \tag{1.9}$$

Où P_r et P_t représentent la puissance reçue et la puissance de transmission.
G_t et G_r représentent respectivement le gain de transmission et le gain de réception de l'antenne.
Le terme $\left(\frac{\lambda}{4\pi R}\right)^2$ est appelé facteur de perte en espace libre avec λ la longueur d'onde et R la distance entre les deux antennes. Enfin n représente les différentes pertes liées à la désadaptation (réflexion et polarisation).

1.4.6.2 Modèle Multi-Wall (MWM)

Un modèle plus sophistiqué a été développé par Motley et Keenan [20]. Ce modèle prend en compte tous les murs et planchers pénétrés par des pertes de pénétration individuelles en fonction de leur épaisseur et de leur matériau. Ce modèle de propagation est le plus approprié aux environnements intérieurs (indoor). Il prend en compte les atténuations dues à la pénétration de l'onde entre l'émetteur et le récepteur dans les obstacles. Le modèle est représentée par l'équation suivante :

$$PL_{MWF}[dB] = FSL + 10 * n * \log_{10}(d) + \sum_{i=1}^{I} \sum_{k=1}^{K_{wi}} (L_{wik}) + \sum_{j=1}^{J} \sum_{k=1}^{K_{fj}} (L_{fjk}) \tag{1.10}$$

$$RSSI[dBm] = P_{TX}[dBm] - PL_{MWF}[dBm] \tag{1.11}$$

Où :

-FSL : désigne Free-Space Losses ;

-L_{wik} : est l'atténuation du $k^{ème}$ mur de la catégorie i ;

-L_{fjk} : est l'atténuation du $k^{ème}$ étage de la catégorie j ;

-K_{wi} : est le nombre des murs de la catégorie i ;

-K_{fj} : est le nombre d'étages de la catégorie j.

1.4.7 Durée de vie

En effet, les nœuds capteurs et routeurs étant principalement alimentés par des batteries ont de ce fait une durée de vie limitée qui dépend principalement de la consommation d'énergie. Par conséquent, l'utilisation appropriée de cette source d'énergie est essentielle pour prolonger la durée de vie du réseau [21]. D'ailleurs, les composants radiofréquences sont plus gourmands en énergie que l'unité de traitement de données. La durée de vie sera donc déterminée par la consommation en énergie de l'unité de communication qui dépend de la distance séparant les nœuds et du nombre de paquets à transmettre. De nombreuses techniques telles que l'ordonnancement du sommeil, la minimisation des messages de diffusion, la transmission des données par le biais d'un coordinateur, l'ordonnancement du temps pour la collecte et la transmission des données, etc. sont proposées par les chercheurs pour prolonger la durée de vie du réseau.

1.5 Stratégies de déploiement

Le déploiement d'un RCSF peut être vu comme un problème d'optimisation qui consiste à la recherche et la détermination des "variables de décision", soit en les minimisant ou les maximisant, c'est à dire selon le type des contraintes du problème traité. Dans notre cas, ces contraintes sont les critères de déploiement mentionné auparavant (coût, taux de couverture, de connectivité et de sur-couverture). Pour résoudre ce genre de problèmes, de nombreuses méthodes ont été proposées et développées, nous distinguons deux grandes familles : les méthodes déterministes et les méthodes non-déterministes autrement appelées stochastiques, comme nous pouvons voir sur la Figure 1.4.

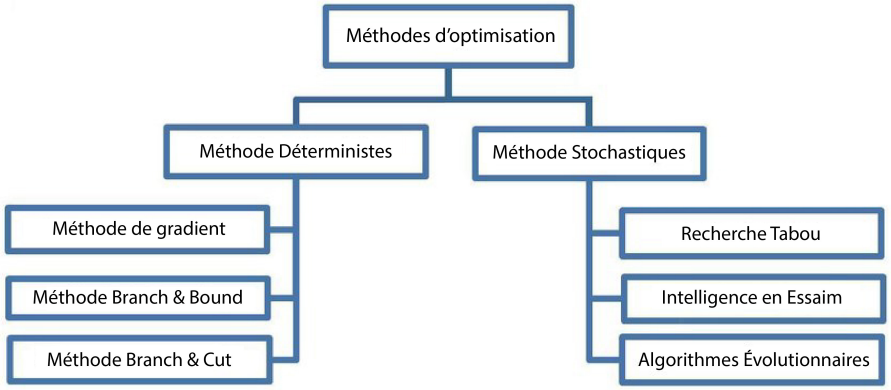

FIGURE 1.4: classification des méthodes d'optimisation

1.5.1 Méthodes d'optimisation déterministes

D'après Alain Berro [22] , ce sont des méthodes qui explorent le domaine de recherche des solutions d'une manière méthodique, les paramètres d'exécution de ces méthodes (les solutions initiales, la direction de recherche,...) sont déjà déterminés d'où vient le nom méthodes déterministes. Ces méthodes restent toujours moins performantes quand il s'agit des problèmes de grande taille, l'inconvénient majeur de ce type de méthode est le temps de calcul qui augmente exponentiellement en fonction de la complexité et la taille du problème. Parmi ces méthodes nous allons présenter dans ce qui suit les plus connues :

1.5.1.1 Méthode de gradient

La méthode de gradient [23] est parmi les méthodes le plus anciennes qui permettent la résolution de problèmes non-linéaires et l'optimisation des fonctions réelles, elle se base sur la connaissance de la dérivée de la fonction objectifs en tous points de l'espace de recherche.

1.5.1.2 Méthode du simplexe

Cette méthode à solutions multiples développée par John A Nelder et Roger Mead [24] est parmi les méthodes les plus courantes d'optimisation locale. En effet, ce qui diffère cette méthode des autres est que l'algorithme améliore un ensemble de solutions appelées polytope au lieu d'une seule solution pendant une itération.

1.5.1.3 Branch Bound

L'idée de cet algorithme est de diviser le problème P en plusieurs sous-problèmes P_1, P_2, P_3..., si les sous-problèmes sont aussi difficiles à résoudre ils seront divisés aussi comme montre la figure 1.5. Il existe un variant plus utilisé de cet algorithme qui s'appelle Branch cut [25].

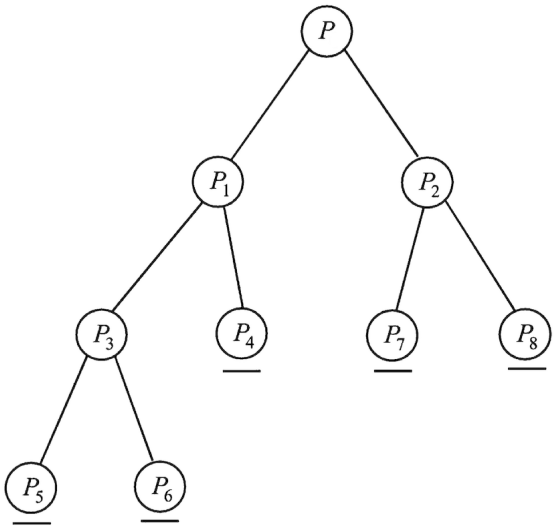

FIGURE 1.5: La division du problème P en plusieurs sous-problèmes par l'algorithme Branch Bound

Les méthodes déterministes restent toujours moins performantes quand il s'agit des problèmes de grande taille, l'inconvénient majeur de ce type de méthode est le temps de calcul qui augmente exponentiellement en fonction de la complexité et la taille du problème. D'où ça semble logique d'essayer de résoudre ces problèmes avec des méthodes d'optimisation non-déterministe aussi appelées stochastiques qui trouvent des solutions s'approchant de l'optimale dans des délais plus courts. La partie suivante est une présentation des méthodes approchées les plus connues et utilisées dans le domaine de l'optimisation.

1.5.2 Méthodes d'optimisation stochastiques

Les méthodes d'optimisation stochastique (SO) génèrent et utilisent des variables aléatoires. Dans notre cas, nous s'intéressons aux méta-heuristiques.

Méta-heuristiques : se sont les méthodes non-déterministes les plus connus qui utilisent des notions de probabilités et des mécanismes aléatoires visant à résoudre des problèmes d'optimisation difficile pour lesquels on ne connaît pas de méthode classique plus efficace.

Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution. Il existe un grand nombre de métaheuristiques différentes, allant de la simple recherche locale à des algorithmes complexes de recherche globale. Dans cette partie nous allons expliquer les méthodes les plus utilisées en donnant des exemples de leur fonctionnement.

1.5.2.1 Recherche Tabou

Proposée par Fred Glover [26], l'idée de cette méthode principale consiste à garder en mémoire l'historique des solutions ou des caractéristiques de solutions déjà explorées. Elle consiste en la création d'une liste qui contiendra toutes les solutions récemment explorées qui deviendront tabous. Ainsi, l'algorithme ne pourra plus les choisir comme solution. Ceci permet à l'algorithme de ne pas retomber dans l'optimum local duquel il vient de sortir. Ainsi à chaque itération l'algorithme choisit le meilleur voisin non tabou même si celui-ci détériore la valeur de la fonction objectif. Ceci permet une meilleure exploration de l'espace de recherche.

1.5.2.2 Intelligence en Essaim

L'optimisation par essaims est une métaheuristique d'optimisation, inventée par Russel Eberhart et James Kennedy en 1995 [27]. Ces algorithmes s'appuient sur le concept d'auto-organisation ou un groupe d'individus peu intelligents peut posséder une organisation globale complexe. Les algorithmes de colonies de fourmis sont les méthodes d'intelligence en essaim les plus utilisées. Cette méthode a montré son efficacité dans la solution du problème du voyageur de commerce. Pour chaque étape, l'individu choisit de passer d'une ville à une autre en fonction de quelques critères :

- Chaque ville est visité une seule fois seulement ;
- Les villes lointaines ont moins de chance d'être choisies car elle sont moins visibles pour l'individu, plus une ville est loin, moins elle a de chance d'être choisie ;
- plus l'intensité de la piste de phéromone déposée sur l'arête entre deux villes est grande, plus le trajet aura de chance d'être choisi ;
- une fois son trajet terminé, l'individu dépose de phéromones sur l'ensemble des arêtes parcourues, si le trajet est court il dépose plus de phéromones ;

- Pour chaque itération de l'algorithme les pistes de phéromones s'évaporent

La figure 1.6 illustre le fonctionnement d'un algorithme de colonies de fourmis appliqué pour le problème du voyageur de commerce :

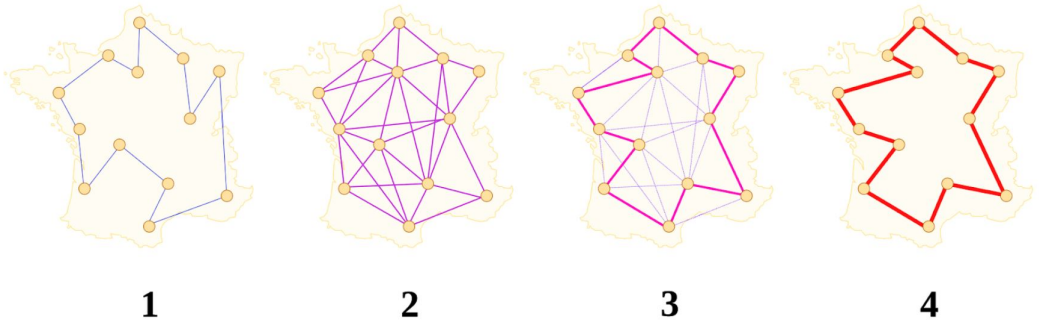

FIGURE 1.6: Utilisation d'un algorithme de colonies de fourmis pour le problème du voyageur de commerce

1.5.2.3 Algorithmes Évolutionnaires

Les algorithmes évolutionnaires (AE) font partie de la famille d'algorithmes bio-inspirés dans le domaine de l'optimisation. Ce type d'algorithmes particulièrement est inspiré de la théorie de l'évolution proposée par Charles Darwin en 1807 [28], l'idée principale est d'assimiler les phénomènes de sélection naturelle et la mutation génétique qui selon Darwin sont la cause de l'apparition des individus qui sont plus adaptés à l'environnement. Pour cela l'algorithme fait évoluer une population d'individus qui sont des solutions possibles de la problématique traitée. La figure1.7 montre le déroulement d'un algorithme évolutionnaire classique.

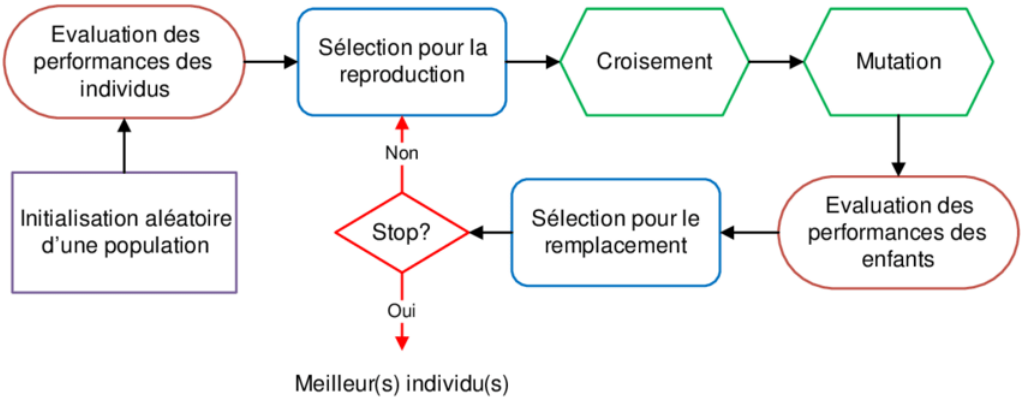

FIGURE 1.7: Le fonctionnement des algorithmes évolutionnaires

Plusieurs variétés plus adaptées aux problématiques traitées ont été proposées. Le principe de fonctionnement reste le même pour tous ces types d'algorithmes évolutionnaires qui

diffèrent seulement dans la structure du génotype des individus ou dans les opérateurs utilisés. Il y a plusieurs types d'algorithmes évolutionnaires :

Les stratégies d'évolution Ces algorithmes permettent la résolution des problèmes d'optimisation continus. La stratégie d'évolution [29] est appliquée aux populations de parents, parmi lesquels des individus sont sélectionnés aléatoirement pour générer des descendants. Ils sont ensuite modifiés par des mutations, qui consistent à ajouter une valeur générée aléatoirement en fonction de la densité de probabilité paramétrée. Les paramètres de la fonction de densité de probabilité, appelés paramètres de stratégie, évoluent également dans le temps selon les mêmes principes que les paramètres qui caractérisent les individus.

Programmation évolutionnaire La programmation évolutionnaire [30] a été initialement conçue dans le but de faire évoluer des machines à états finis et a été par la suite étendue aux problèmes d'optimisation de paramètres. Cette approche met l'emphase sur la relation entre les parents et leurs descendants plutôt que de simuler des opérateurs génétiques d'inspiration naturelle. Contrairement aux autres algorithmes évolutionnaires classiques, la programmation évolutionnaire n'utilise pas une représentation spécifique mais plutôt un modèle évolutionnaire de haut niveau, jumelé à une représentation et à un opérateur de mutation directement appropriés au problème à résoudre.

Programmation génétique La programmation génétique [31] et les algorithmes génétiques [32] sont les AEs les plus connus et les plus utilisés. La programmation génétique particulièrement est caractérisée par la représentation des individus sous la forme d'un arbre.

Algorithme génétiques Les algorithmes génétiques (AG) sont une technique de recherche et d'optimisation heuristique inspirée de l'évolution naturelle. Ils ont été appliqués avec succès à un large éventail de problèmes du monde réel d'une grande complexité. Ce sont des AEs ou la représentation des individus sous forme de vecteur binaire ou chaine de caractère.

1.6 Conclusion

Ce premier chapitre a commencé par une présentation des bâtiments intelligents et leurs fonctionnement dans les différents domaines, nous avons vu aussi la notion des RCSFs ainsi que les applications et performances de cette nouvelle technologie, nous avons également

réalisé une étude sur les différentes modélisations possibles des critères de déploiement ayant une influence sur les performances du RCSF. Enfin nous avons exposé quelques méthodes utilisées pour la résolution du problème de déploiement des RCSFs. Mais vu le degré de complexité de cette problématique et en se basant sur plusieurs travaux dans ce domaine, nous sommes arrivés à la conclusion que les méthodes déterministes ne sont pas capables de résoudre ce problème, pour cela notre choix s'est porté sur les méthodes stochastiques (non-déterministes) et plus précisément les métaheuristiques. Dans le prochain chapitre, nous allons utiliser l'algorithme génétique pour l'optimisation multi-objectif et l'adapter à notre problématique.

Chapitre 2

Approche évolutionnaire multi-objectif pour le déploiement d'un RCSF

2.1 Introduction

S'inspirant de la théorie de l'évolution naturelle de Charles Darwin, l'une des techniques les plus fascinantes de résolution de problèmes est la famille d'algorithmes appelée "algorithme évolutionnaire". Au sein de cette famille, la branche la plus importante et la plus utilisée est connue sous le nom d'algorithmes génétiques (AGs). Ces derniers attirent l'attention des chercheurs pour résoudre les problèmes liés à la couverture et à la connectivité et à d'autres critères du déploiement d'un RCSF.

Dans ce chapitre, nous présenterons l'**AG** et son analogie avec l'évolution darwinienne, et nous nous plongerons dans le principe de fonctionnement de base ainsi que les différences entre l'**AG** et les algorithmes traditionnels et couvrirons leurs avantages, leurs limites et leurs utilisations. Ensuite, nous allons nous concentrer sur l'adaptation de ces algorithmes à notre problématique en tenant en compte des différents modèles expliqués dans le chapitre précédent concernant les critères de déploiement voir le coût, la connectivité, la couverture et la sur-couverture. Nous allons également procéder à plusieurs simulations pour les différents AGs utilisés dans le but de déterminer les valeurs optimales des paramètres des différents algorithmes, leurs performances, leurs temps d'exécution et la qualité des solutions obtenus.

2.2 Algorithmes génétiques

2.2.1 Principes de l'évolution darwinienne

Les principes de la théorie de l'évolution darwinienne [28] peuvent être résumés à l'aide des principes suivants :

2.2.1.1 Diversité

Les traits (attributs) des individus appartenant à une population peuvent varier. Par conséquent, ces individus diffèrent les uns des autres dans une certaine mesure, par exemple, dans leur comportement ou leur apparence.

2.2.1.2 Hérédité

Certains traits sont systématiquement transmis des individus à leur progéniture. Par conséquent, les descendants ressemblent davantage à leurs parents.

2.2.1.3 Sélection naturelle

Les populations se battent généralement pour les ressources dans leur environnement donné. Les individus possédant les traits les mieux adaptés à l'environnement auront plus de chances de survivre et contribueront également à une plus grande descendance à la génération suivante. En d'autres termes, l'évolution maintient ceux qui sont mieux adaptés à leur environnement et qui ont plus de chances de survivre, de se reproduire et de transmettre leurs caractéristiques à la génération suivante. En outre, un facteur important de l'évolution est le croisement ou la recombinaison, qui permet de créer une progéniture avec un mélange des caractéristiques de ses parents. Le croisement contribue à maintenir la diversité de la population et à rassembler les meilleurs traits au fil du temps. De plus, les mutations peuvent jouer un rôle dans l'évolution en introduisant des variations aléatoires sur les caractéristiques d'un individu.

2.2.2 Différents composants des algorithmes génétiques

En imitant le processus de sélection et de reproduction naturelles, les AGs peuvent produire des solutions de haute qualité pour divers problèmes de recherche, d'optimisation et d'apprentissage. Dans les sections suivantes, nous allons décrire les différents composants des AGs qui permettent cette analogie avec l'évolution darwinienne.

2.2.2.1 Chromosome

Dans la nature, si deux individus se reproduisent pour créer une progéniture, chaque chromosome de la progéniture portera un mélange de gènes des deux parents. En imitant ce concept, dans le cas des algorithmes génétiques, chaque individu est représenté par un chromosome représentant une collection de gènes. Par exemple, un chromosome peut être exprimé comme une chaîne binaire, où chaque bit représente un seul gène. la Figure 2.1 montre un exemple d'un tel chromosome à code binaire.

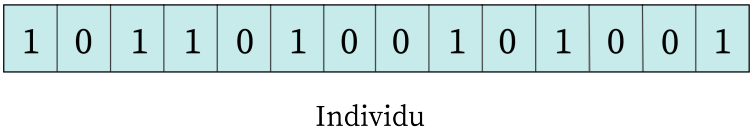

FIGURE 2.1: Chromosome codé en binaire

2.2.2.2 Population

À tout moment, les algorithmes génétiques maintiennent une population d'individus, c'est-à-dire une collection de solutions candidates pour le problème à résoudre. La population représente continuellement la génération actuelle et évolue dans le temps lorsque la génération actuelle est remplacée par une nouvelle. Comme chaque individu est représenté par un chromosome, cette population d'individus peut être considérée comme une collection de chromosomes, voir Figure 2.2 :

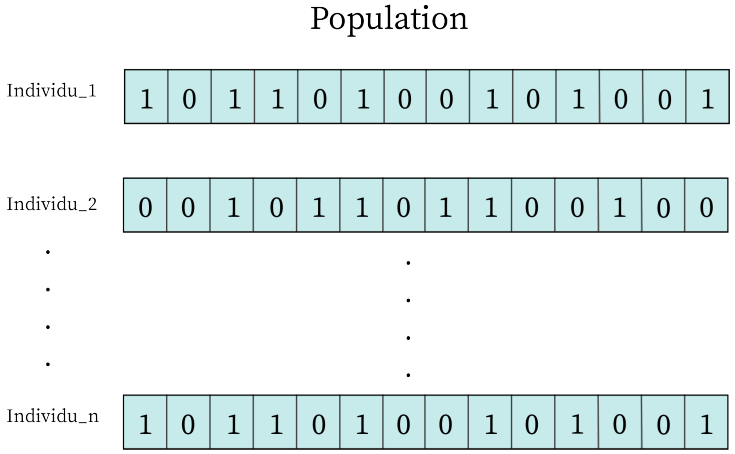

FIGURE 2.2: Population des chromosomes codés en binaire

2.2.2.3 Fonction fitness

À chaque itération de l'algorithme, les individus sont évalués à l'aide d'une fonction fitness. Il s'agit de la fonction que nous cherchons à optimiser ou du problème que nous tentons de résoudre. Les individus qui obtiennent un meilleur score de fitness représentent de meilleures solutions et sont plus susceptibles d'être choisis pour se reproduire et être représentés dans la génération suivante. Au fil du temps, la qualité des solutions s'améliore, les valeurs de fitness augmentent, et le processus peut s'arrêter dès qu'une solution est trouvée avec une valeur de fitness satisfaisante.

2.2.2.4 Opérateurs de sélection

Après avoir calculé la valeur fitness de chaque individu de la population, un processus de sélection est utilisé pour déterminer quels individus de la population se reproduiront et créeront la progéniture qui formera la génération suivante. L'opérateur de sélection est chargé de sélectionner les individus de la population actuelle d'une manière qui donne un avantage aux meilleurs individus. Ce processus est basé sur le score de fitness des individus.

Ceux dont le score est le plus élevé ont plus de chances d'être choisis et de transmettre leur matériel génétique à la génération suivante. Les individus dont la valeur d'aptitude est faible peuvent toujours être choisis, mais avec une probabilité moindre. Par la suite, nous citerons les méthodes de sélection les plus connues :

La roue de roulette La probabilité de sélection d'un individu est directement proportionnelle à sa valeur fitness. Ceci est comparable à l'utilisation d'une roue de roulette et à l'attribution à chaque individu d'une partie de la roue proportionnelle à sa valeur fitness. Lorsque la roue est tournée, les chances de chaque individu sélectionné sont proportionnelles à la taille de la partie de la roue qu'elle occupe. Chaque fois que l'on tourne la roue, le point de sélection est utilisé pour choisir un seul individu parmi l'ensemble de la population. La roue est ensuite tourner à nouveau pour sélectionner l'individu suivant jusqu'à ce que nous ayons suffisamment d'individus sélectionnés pour remplir la prochaine génération. Par conséquent, le même individu peut être choisi plusieurs fois, la roue de roulett est montrée sur la Figure2.3.

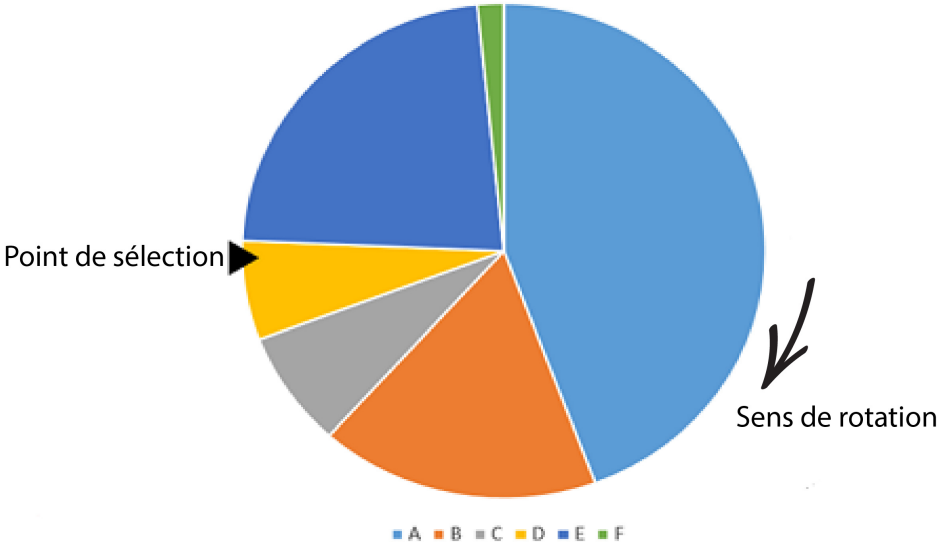

FIGURE 2.3: Roue de roulette

Sélection par tournoi À chaque tour de cette méthode, deux individus ou plus sont choisis au hasard dans la population, et celui avec la valeur fitness la plus élevée l'emporte et est sélectionné. Un aspect intéressant de cette méthode de sélection est que, tant que nous pouvons comparer deux individus quelconques et déterminer lequel d'entre eux est le meilleur, la valeur réelle de la fonction de fitness n'est pas nécessaire pour la comparaison, le principe de cette méthode est montré dans la Figure2.4.

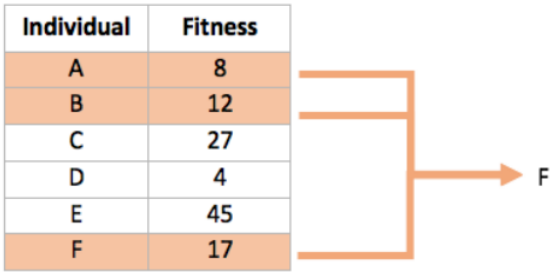

FIGURE 2.4: Méthode de sélection par tournoi

Élitisme Alors que la valeur fitness moyenne de la population d'algorithmes génétiques augmente généralement au fil des générations, il est possible à tout moment que les meilleurs individus de la génération actuelle soient perdus. Cela est dû au fait que les opérateurs de sélection, de croisement et de mutation modifient les individus dans le processus de création de la génération suivante. Dans de nombreux cas, la perte est temporaire car ces individus seront réintroduits dans la population dans une génération future. Cependant, si nous voulons garantir que les meilleurs individus parviennent toujours à la génération suivante, nous pouvons appliquer la stratégie d'élitisme facultative. Cela signifie que les n premiers individus sont dupliqués dans la génération suivante avant d'entamer la procédure de sélection, du croisement et de la mutation. Les individus d'élite qui ont été dupliqués sont toujours élus pour le processus de sélection afin qu'ils puissent toujours être utilisés comme parents de nouvelles personnes. L'élitisme peut parfois avoir un impact positif significatif sur les performances de l'algorithme car il évite le gaspillage de temps nécessaire pour redécouvrir les bonnes solutions qui ont été perdues auparavant.

2.2.2.5 Opérateurs de croisement

L'opération de croisement se fait généralement en prenant deux individus à la fois et en interchangeant des parties de leurs chromosomes pour créer deux nouveaux chromosomes représentant la progéniture. Cet opérateur est généralement appliqué avec une valeur de probabilité élevée. Si le croisement n'est pas appliqué, les deux parents sont directement clonés dans la génération suivante. Les sections suivantes décrivent certaines des méthodes de croisement couramment utilisées et leurs cas d'utilisation typiques.

Croisement à point unique Un emplacement sur les chromosomes des deux parents est sélectionné au hasard. Cet emplacement est appelé point de croisement ou point de coupure. Les gènes à droite de ce point sont échangés entre les deux chromosomes parents.

En conséquence, nous avons deux descendants, où chacun d'eux porte des informations génétiques des deux parents. la figure 2.5 montre une opération de croisement en un seul point menée sur une paire de chromosomes binaires, le point de croisement étant situé entre les cinquième et sixième gènes.

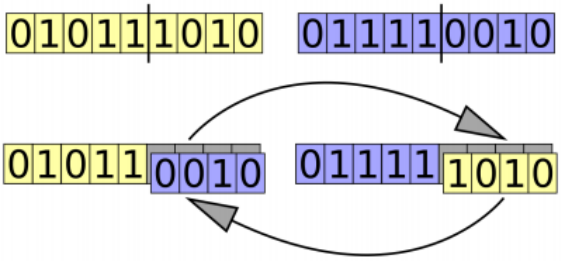

FIGURE 2.5: Croisement à point unique

Croisement à deux points Dans la méthode de croisement à deux points, deux points de croisement sur les chromosomes des deux parents sont sélectionnés au hasard. Les gènes résidant entre ces points sont échangés entre les deux chromosomes parents. la figure 2.6 montre un croisement en deux points effectué sur une paire de chromosomes binaires, le premier point de croisement étant situé entre les troisième et quatrième gènes, et l'autre entre les septième et huitième gènes.

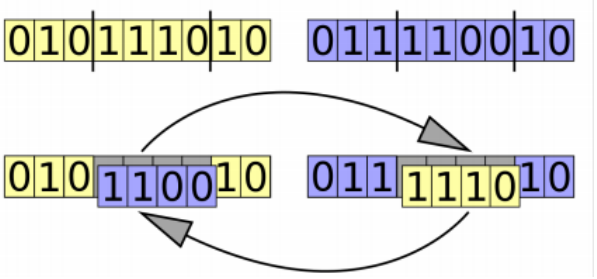

FIGURE 2.6: Croisement à deux points

2.2.2.6 Opérateurs de mutation

Les mutations sont mises en œuvre sous la forme de changements aléatoires d'une ou plusieurs valeurs chromosomiques (gènes) de chaque individu nouvellement créé, par exemple en inversant un bit dans une chaîne binaire. La mutation se produit généralement avec une très faible probabilité.

Mutation d'un bit Un gène est sélectionné au hasard et sa valeur est inversée, comme indiqué dans la figure 2.7 :

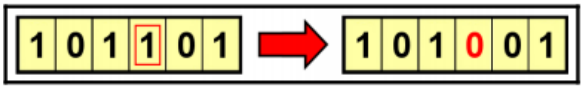

FIGURE 2.7: Mutation d'un bit

Mutation d'échange Lors de l'application de la mutation d'échange à des chromosomes binaires ou à base d'entiers, deux gènes sont sélectionnés au hasard et leurs valeurs sont permutées, comme indiqué dans la figure 2.8 :

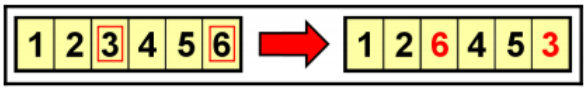

FIGURE 2.8: Mutation d'échange

2.2.3 Principales étapes d'un algorithme génétique de base

2.2.3.1 Création de la population initiale

La population initiale est un ensemble de solutions possibles (appelés individus) choisies au hasard. Comme les algorithmes génétiques utilisent un chromosome pour représenter chaque individu, la population initiale est en fait un ensemble de chromosomes qui doivent se conformer au format que nous avons choisi pour le problème à résoudre, par exemple, des chaînes binaires d'une certaine longueur.

2.2.3.2 Calcul de la valeur de fitness

La valeur de la fonction de fitness est calculée pour chaque individu. Cette opération est effectuée une fois pour la population initiale, puis pour chaque nouvelle génération après l'application des opérateurs génétiques. Comme la valeur de fitness de chaque individu est indépendante des autres, ce calcul peut être effectué simultanément. Étant donné que l'étape de sélection qui suit le calcul de la valeur de fitness considère généralement les individus ayant un score plus élevé comme de meilleures solutions, les algorithmes génétiques sont naturellement orientés vers la recherche d'un score maximal. Dans le cas d'un problème où la valeur minimale est souhaitée, le calcul de la valeur de fitness doit être inverser, par exemple en la multipliant par une valeur de (-1).

2.2.3.3 Application de la sélection, du croisement et de la mutation

L'application des opérateurs génétiques de sélection, de croisement et de mutation à la population entraîne la création d'une nouvelle génération basée sur de meilleurs individus que les individus actuels.

2.2.3.4 Vérification des conditions d'arrêt

Il peut y avoir plusieurs conditions à vérifier pour déterminer si le processus peut s'arrêter. Les conditions d'arrêt les plus couramment utilisées sont les suivantes : - Un temps prédéterminé s'est écoulé depuis le début du processus.

- Un certain coût ou budget a été consommé, tel que le temps CPU ou la mémoire.

- Un nombre maximal de générations a été atteint, cela permet également de limiter la durée d'exécution et les ressources informatiques consommées par l'algorithme.

- Il n'y a pas eu d'amélioration notable au cours des dernières générations, cela peut être mis en œuvre en stockant la meilleure valeur de fitness obtenue à chaque génération et en comparant la meilleure valeur actuelle à celle obtenue un nombre prédéfini de générations auparavant. Si la différence est inférieure à un certain seuil, l'algorithme peut s'arrêter.

2.2.4 Différences par rapport aux algorithmes traditionnels

Il existe plusieurs différences importantes entre les algorithmes génétiques et les algorithmes traditionnels de recherche et d'optimisation comme Monte-Carlo et la recherche Tabou. En effet, l'analogie avec l'évolution naturelle permet de surmonter les obstacles rencontrés par une approche traditionnelle, en particulier pour les problèmes comportant un grand nombre de paramètres ou des représentations mathématiques assez complexes. Les principales caractéristiques des algorithmes génétiques qui les distinguent des autres algorithmes sont les suivantes : le maintien d'une population de solutions, l'utilisation d'une représentation génétique des solutions, l'utilisation du résultat d'une fonction fitness et la manifestation d'un comportement probabiliste.

2.2.4.1 Ensemble de solutions

Les **AG**s sont capables de résoudre des problèmes extrêmement vastes dont l'espace de recherche est important. Ils sont très utiles pour naviguer dans de vastes espaces de recherche et trouver rapidement des solutions quasi optimales de manière heuristique car la recherche de solution est menée sur une population de solutions candidates (individus) plutôt que sur

un seul candidat. En revanche, la plupart des autres algorithmes traditionnels conservent une solution unique et la modifient de manière itérative à la recherche de la meilleure solution.

2.2.4.2 Représentation génétique

Les chromosomes nous permettent de faciliter les opérations génétiques de croisement et de mutation. De plus, les algorithmes génétiques ne sont pas conscients de ce que les chromosomes représentent et ne tentent pas de les interpréter.

2.2.4.3 Flexibilité de la fonction fitness

Contrairement à de nombreux algorithmes de recherche traditionnels, les algorithmes génétiques ne prennent en compte que la valeur obtenue par la fonction de fitness et ne s'appuient pas sur les dérivés ou toute autre information complexe. Cela les rend aptes à traiter des fonctions qui sont difficiles ou impossibles à différencier mathématiquement.

2.2.4.4 Comportement probabiliste

Alors que de nombreux algorithmes traditionnels sont déterministes par nature, les règles utilisées par les algorithmes génétiques pour passer d'une génération à la suivante sont probabilistes. Malgré la nature de ce processus, la recherche basée sur les algorithmes génétiques n'est pas aléatoire ; au contraire, elle utilise l'aspect aléatoire pour diriger la recherche vers des zones de l'espace de recherche où il y a une meilleure chance d'améliorer les résultats.

2.2.5 Avantages des algorithmes génétiques

Les caractéristiques uniques des algorithmes génétiques dont nous avons parlé dans les sections précédentes offrent plusieurs avantages par rapport aux algorithmes de recherche traditionnels. Les principaux avantages des algorithmes génétiques sont les suivants :

2.2.5.1 Capacité d'optimisation globale

Bien que les **AG**s ne garantissent pas l'optimalité, ils sont généralement utiles dans la pratique. Bhandar et al. [33] ont montré que, bien que les **AG**s ne garantissent pas la convergence vers une solution optimale, ils évitent les optima locaux avec une forte probabilité. En effet, la plupart des algorithmes traditionnels ont tendance à rester bloqués dans un maximum local plutôt que de trouver le maximum global. Les algorithmes génétiques, en revanche, sont moins sensibles à ce phénomène et ont plus de chances de trouver le maximum

global. Cela est dû à l'utilisation d'une population de solutions candidates plutôt qu'une seule, et aux opérations de croisement et de mutation qui vont, dans de nombreux cas, aboutir à des solutions candidates éloignées des précédentes. Cela reste vrai tant que l'on parvient à maintenir la diversité de la population. L'image suivante illustre les différences entre les points maxima et minima globaux et locaux :

2.2.5.2 Adaptés aux problèmes complexes

Étant donné que les **AG**s ne requièrent que le résultat de la fonction de fitness pour chaque individu et qu'ils ne se préoccupent pas des autres aspects, tels que les dérivées, ils peuvent être utilisés pour des problèmes comportant des représentations mathématiques complexes ou des fonctions difficiles ou impossibles à différencier [14]. Par conséquent, les **AG**s peuvent fonctionner même lorsque la fonction objective n'est pas exactement connue, car ils se basent uniquement sur l'évaluation d'une fonction objective (sans nécessairement connaître la fonction objective de manière explicite).

2.2.5.3 Prise en charge de l'optimisation multiobjectif

De nombreux problèmes pratiques nécessitent l'optimisation de plusieurs contraintes qui peuvent potentiellement avoir une dépendance les uns avec les autres. Un avantage important des **AG**s est qu'ils peuvent facilement prendre en charge l'optimisation conjointe de plusieurs objectifs[34].

2.3 Algorithmes génétiques multi-objectifs

Il est clair que les **AG**s à objectif unique (AGSO) sont populaires lorsqu'il s'agit d'optimiser un objectif unique avec une fonction de fitness scalaire car ils sont moins intensifs sur le plan informatique. En revanche, dans une optimisation multi-objectif, il y a plusieurs objectifs qui doivent être optimisés simultanément. Dans un tel scénario, une solution unique peut ne pas convenir à plusieurs objectifs. Une solution peut être la meilleure pour un objectif mais la pire dans d'autres cas. L'espace de recherche est optimisé, dans toutes les directions, par des solutions multiples. Ces catégories de solutions sont des solutions non dominées qui se trouvent sur le front de Pareto [35]. Chaque point du front de Pareto est optimal, car il n'existe aucune solution unique susceptible d'améliorer un vecteur objectif sans dégrader au moins un autre objectif. Presque toutes les solutions situées sur le front de Pareto seraient des

compromis par rapport à des objectifs multiples. Les **AG**s multi-objectifs (AGMO) peuvent être utilisés dans presque tous les contextes d'optimisation, à condition que le codage des chromosomes et la fonction objectif soient pris en compte de manière appropriée. la figure 2.9 montre le Front de Pareto d'une fonction bi-objectifs.

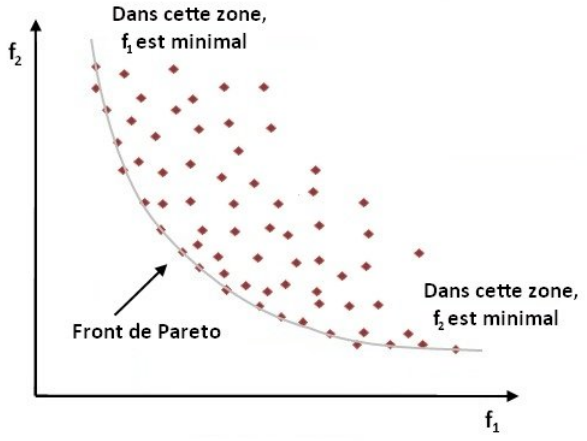

FIGURE 2.9: Front de Pareto d'une fonction bi-objectifs

Il existe plusieurs variantes des AGs multi objectifs mais dans notre cas, nous nous intéressons à ceux qui utilisent la dominance de Pareto. Dans la prochaine section nous allons présenter deux variantes SPEA-II et NSGA-II

2.3.0.1 SPEA-II

Strength Pareto Evolutionary Algorithm II est un algorithme génétique multi-objectif (AGMO) qui est une extension de SPEA [36]. La famille des AGMO SPEA intègre la notion de dominance selon Pareto dans la phase de sélection. Les nouveautés introduites dans cet algorithme par rapport a l'algorithme génétique classique sont la création d'une archive dans laquelle l'algorithme stocke toutes les solutions non dominées a chaque itération et enfin la réduction du nombre de solution dans l'archive à l'aide d'une méthode de clustering, ses étapes sont montrés sur la Figure2.10.

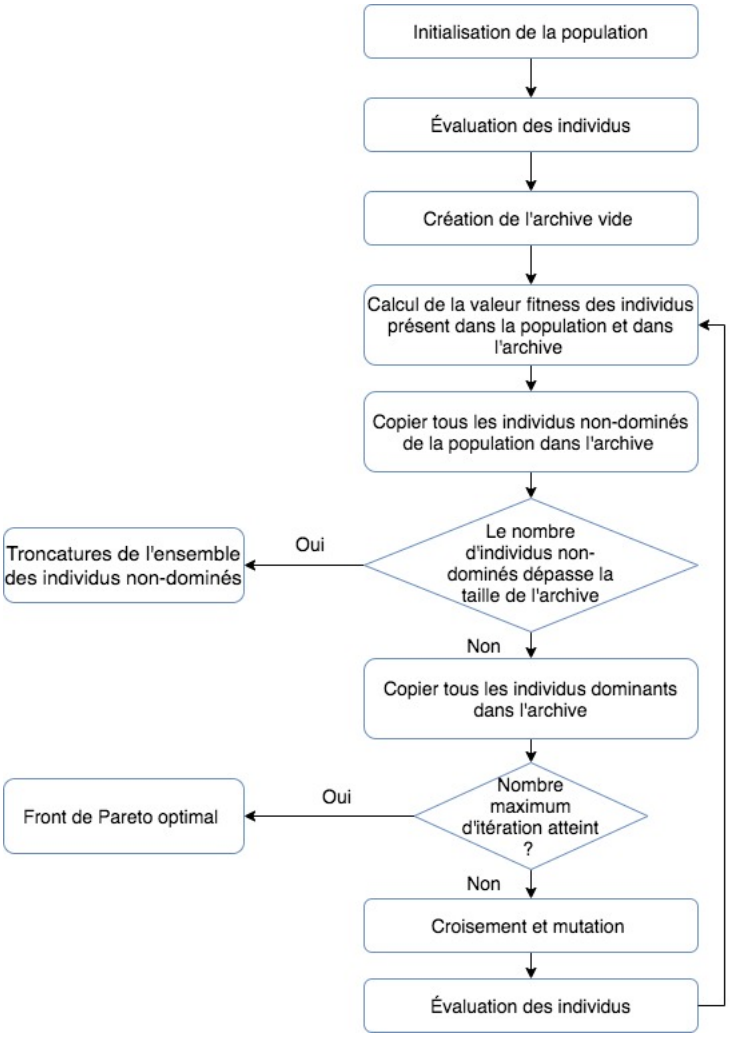

FIGURE 2.10: Principe de fonctionnement de l'algorithme SPEA-II.

2.3.0.2 NSGA-II

NSGA-II est un AGMO basé sur le principe de dominance selon Pareto [37], les individus sont classés selon la notion de dominance de Pareto. L'algorithme regroupe les individus non-dominés dans le premier front. Après ceci, d'autres fronts sont formés en ignorant les individus précédemment choisis. Cette opération est répétée d'une manière récursive jusqu'à ce qu'il ne reste plus d'individus dans la population. La Figure 2.11 montre le fonctionnement de NSGA-II.

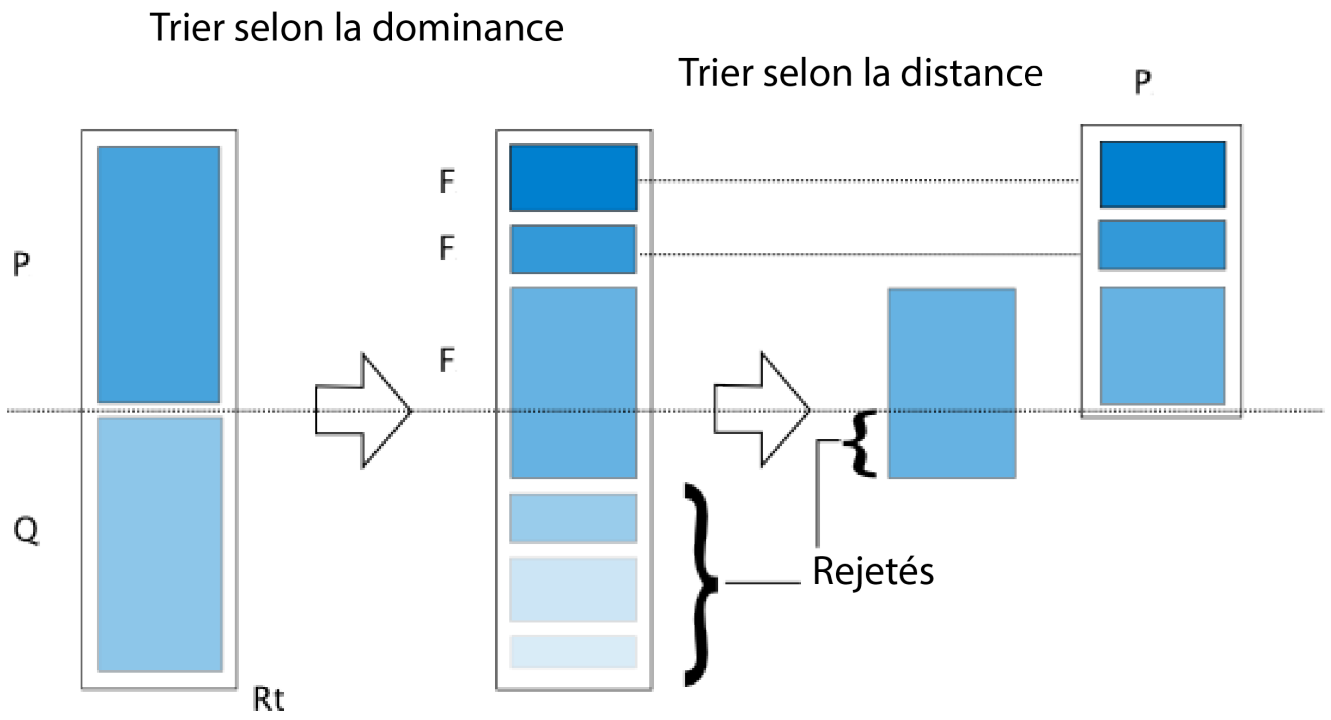

FIGURE 2.11: Principe de fonctionnement de l'algorithme NSGA-II.

2.4 Travaux récents

Les (RCSF) sont un domaine de recherche en évolution continue avec une multitude de contextes d'application. Le déploiement des nœuds capteurs est une phase décisive qui influe considérablement sur le fonctionnement et la performance du réseau. Nous présentons tout d'abord quelques travaux de recherche les plus récents qui concernent les méthodologies de résolution de cette problématique :

Les auteurs dans [38] présentent un algorithme génétique qui recherche une solution au problème des trous de couverture dans le réseau. L'algorithme proposé détermine le nombre minimum et les meilleurs emplacements des nœuds mobiles qui doivent être ajoutés après le déploiement initial des nœuds fixes. La performance de l'algorithme génétique proposé a été évaluée à l'aide de plusieurs indicateurs, et les résultats de simulation montrent que cet algorithme permet d'optimiser la couverture du réseau en termes de ratio de couverture global et de nombre de nœuds mobiles supplémentaires.

Les travaux de [39] étudient les différentes approches multi-objectif pour la résolution de la problématique de déploiement des capteurs suivant différents paramètres (couverture, scalabilité, connectivité, coût, durée de vie, latence). Les auteurs présentent dans cette étude, les travaux basés sur les algorithmes génétiques et ceux basés sur les essaims particulaires. Ils présentent également les différents environnements de simulation dans le cas multi-objectif. Cette simulation est décomposée en deux phases. La première consiste à simuler le comportement des nœuds et les résultats sont optimisés jusqu'à atteindre la convergence. La deuxième phase consiste à alimenter un réseau de simulation par les résultats obtenus pour vérifier la solution trouvée.

Dans [34] Les auteurs présentent et évaluent deux **AE**s pour résoudre le problème du déploiement des RCSF. L'architecture du système proposé permet la planification d'un RCSF pour une application SB (smart building). en tenant compte des contraintes liées au bâtiment qui peuvent affecter le réseau. L'idée de ce travail est le codage de la solution (chromosome), qui peut incorporer plusieurs informations en un seul bit (coordonnées des nœuds et coût du réseau). Une étude comparative entre deux **AE**s (**AG** classique et NSGA-II) a été présentée. Ils ont conclu que le choix de la méthode utilisée dépend du besoin de l'utilisateur, d'après les résultats des simulations l'utilisation de NSGA-II est recommandée.

Hanaa ZeinEldine et al [40] présentent une technique de déploiement dynamique améliorée basée sur l'algorithme génétique (IDDT-GA). Cette technique a été conçue pour maximiser la couverture du réseau en réduisant le nombre de nœuds de capteurs dans le déploiement aléatoire donc elle est une solution prometteuse non seulement pour démontrer la notation du codage à longueur variable par un croisement amélioré à deux points, mais aussi pour garantir la connectivité entre les nœuds de capteurs. Les résultats ont confirmé la supériorité et l'efficacité de la technique proposée, qui présente de meilleurs taux de couverture et un taux de chevauchement minimal par rapport à d'autres méthodes de déploiement.

2.5 Modélisation mathématique du problème de déploiement

Dans le chapitre précédent, nous avons décrit et modélisé plusieurs critères qui ont une influence sur les performances d'un RCSF. Le but principal est de trouver le meilleur emplacement des nœuds tout en respectant les contraintes de coût, connectivité, couverture

et sur couverture. Pour cela nous modélisons la problématique mathématiquement. Comme mentionné au part avant l'espace de déploiement est modélisé avec une grille de L*l cellules de $1m^2$ et les capteurs seront placés au centre des cellules et donc nous avons la matrice de déploiement comme suit :

$$D(i,j) = \begin{cases} 1 & \text{si un capteur ou un relais est déployé dans la cellule ;} \\ 0 & \text{sinon.} \end{cases} \tag{2.1}$$

Nous normalisons le coût afin d'avoir la même unité que les autres objectifs, cela nous permettra de modéliser le coût comme suit :

$$Coût = \frac{\sum_{i=0}^{l-1}\sum_{j=0}^{L-1} D(i,j)}{L * l} * 100 \tag{2.2}$$

Pour le taux de couverture nous avons opté pour le modèle de détection binaire. En utilisant le même modèle de détection binaire nous avons aussi modélisé la sur-couverture qui nous a donné les deux matrices suivantes :

$$Couv(i,j) = \begin{cases} 1 & \text{si la cellule est couverte par au moins un capteurs ;} \\ 0 & \text{Sinon.} \end{cases} \tag{2.3}$$

$$Sur - couv(i,j) = \begin{cases} 1 & \text{si la cellule est couverte par au moins deux capteurs ;} \\ 0 & \text{Sinon.} \end{cases} \tag{2.4}$$

Et donc les taux de couverture et de sur-couverture peuvent être calculés a partir des équations :

$$Couv_{RI} = \frac{\sum_{i=0}^{l-1}\sum_{j=0}^{L-1} Couv(i,j)}{L \cdot l} \cdot 100 \tag{2.5}$$

$$Sur - couv_{RI} = \frac{\sum_{i=0}^{l-1}\sum_{j=0}^{L-1} Sur - couv(i,j)}{L \cdot l} \cdot 100 \tag{2.6}$$

Afin de modéliser la connectivité nous allons adopté le modèle simple de communication binaire expliqué précédemment [41], d'où la matrice de connectivité est définie comme suit :

$$Con(i,j) = \begin{cases} 1 & \text{si la cellule (i,j) est dans le rayon de communication d'au moins 1 capteur ;} \\ 0 & \text{sinon.} \end{cases} \tag{2.7}$$

Pour conclure cette modélisation mathématique, le taux de connectivité est donné par :

$$Con_{RI} = \frac{\sum_i \sum_j Con(i,j)}{L * l} * 100 \tag{2.8}$$

2.6 Adaptation de l'algorithme génétique au problème de déploiement du RCSF

Après avoir validé les modèles à utiliser et formulé le problème de déploiement des RCSF, l'étape suivante consiste à trouver la solution optimale. Dans cette partie du chapitre, nous allons utiliser les algorithmes évolutionnaires que nous avons mentionnés précédemment pour résoudre le problème du déploiement du RCSF, et les adapter à notre problème d'optimisation multiobjectif. Dans un premier temps, nous allons définir un codage afin de représenter le chromosome. Ensuite, nous passerons à l'étape d'initialisation où la population est générée de manière aléatoire.

2.6.1 Codage du chromosome

La solution à notre problème d'optimisation est modélisée sous forme de matrice, qui définit l'espace de déploiement dans lequel notre RCSF est mis en place. Chaque case indique la présence ou l'absence d'un capteur et son emplacement au niveau de la grille. Nous allons donc transformer cette grille en un vecteur binaire, où chaque bit représente un gène, une caractéristique de l'individu. Cette transformation montrée sur la Figure2.12, est nécessaire pour l'utilisation des algorithmes choisis car ils ne peuvent pas manipuler des données de structure matricielle. De ce fait, il faut aussi préciser que la taille du vecteur dépend évidemment des dimensions du bâtiment, autrement dit de l'espace de déploiement.

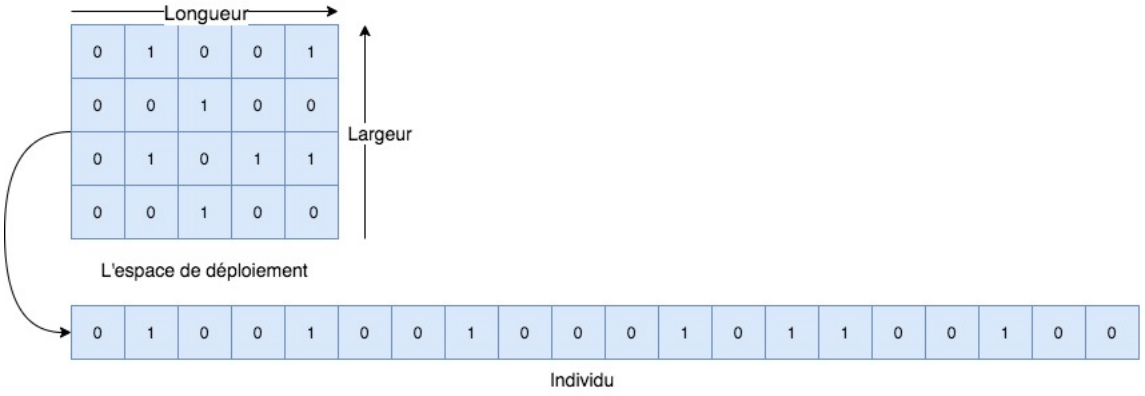

FIGURE 2.12: Représentation de l'espace du déploiement codé en binaire

2.6.2 Création de la fonction d'évaluation

En effet, les différents critères de déploiement que nous avons choisis sont : le coût, qui est proportionnel au nombre de capteurs déployés, la couverture du bâtiment qui doit être maximisée afin d'avoir une bonne portée, la connectivité du réseau, qui doit également être optimisée afin d'assurer une meilleure qualité de service, et enfin, la sur-couverture visant à éviter de déployer des capteurs de manière redondante et sans utilité. A ce niveau, nous allons définir une fonction d'évaluation qui englobe nos quatre fonctions objectives. La fonction d'évaluation aura un individu comme paramètre et produira quatre valeurs, qui représenteront la valeur de fitness pour chaque objectif. Cela signifie que nous aurons plusieurs fonctions de fitness à optimiser simultanément.

2.6.3 Phase d'initialisation

Après avoir défini les dimensions de l'espace de déploiement, il suffit de fixer le nombre d'individus par population. Ensuite, nous allons générer la population initiale d'une manière aléatoire pour préparer l'espace de recherche utilisé par notre algorithme évolutionnaire. En d'autres termes, l'initialisation consiste à déployer quelques nœuds arbitrairement dans l'espace de déploiement. Cette phase est assez rapide à implémenter et ne consomme pas trop de temps lors de l'exécution. Comme nous l'avons décrit précédemment, le principal objectif de l'utilisation de tout un espace de recherche, c'est à dire plusieurs solutions possibles, est de garantir que le principe de diversité est présent dans la population et éviter ainsi que l'algorithme tombe dans un optimum local.

2.6.4 Evaluation des individus

Cette opération se fait à l'aide de la fonction fitness (fonction d'évaluation). Dans notre cas, la fonction d'évaluation évalue le coût de chaque individu, son taux de couverture, son taux de sur-couverture et enfin son taux de connectivité. De plus, nous allons calculer une valeur de fitness (score) associée à chacun des individus pour mieux quantifier la qualité de la solution. Cette valeur est une somme pondérée des quatre objectifs et le signe moins représente la minimisation, tandis que le signe plus représente la maximisation.

$$score = \frac{-Coût + Couv_{RI} + Con_{RI} - Surcouv_{RI}}{4} \tag{2.9}$$

2.6.5 Processus d'évolution

Ensuite, nous allons choisir les opérateurs génétiques à utiliser lors du processus d'évolution. Il faut aussi fixer les probabilités de mutation et de croisement, puis on entre dans la boucle d'évolution. La première phase du processus a pour but de sélectionner les individus qui constitueront la génération suivante (les descendants). Ces descendants sont ensuite exposés aux opérations de croisement et de mutation, puis à l'opération d'évaluation afin d'explorer l'espace de recherche et trouver ainsi de meilleurs individus. La population est remplacée entièrement ou partiellement par les descendants. Une fois que la boucle atteint le nombre de générations (itérations) prédéfini, le processus d'évolution est entièrement achevé et la population finale est formée. En effet, l'opérateur de sélection permet de classer la population finale selon le principe de sélection de chaque algorithme utilisé. On obtient ainsi une liste d'individus avec leur coût, leur taux de couverture, leur taux de connectivité et leur taux de sur-couverture.

2.7 Implémentation de l'architecture proposée

Selon le travail proposé par [18], deux objectifs (le coût et la connectivité) ont été proposés pour le déploiement des noeuds routeurs. En revanche, nous allons nous focaliser seulement sur le déploiement des noeuds capteurs. Nous avons aussi supposé que les capteurs avaient un rayon de couverture égal à 1,5 m et un rayon de connectivité égal à 3 m. Pour établir la disposition des nœuds de capteurs, nous allons dans un premier temps étudier l'influence des paramètres suivants : choix des opérateurs d'évolution, probabilité de mutation, probabilité

de croisement, taille de la population ainsi que le nombre de générations sur la qualité des solutions et le temps d'exécution des algorithmes.

2.7.1 Langage de programmation et librairies utilisées

2.7.1.1 Python

Python est l'un des langages de programmation open source les plus utilisés. Ainsi, grâce à des bibliothèques dédiées, il peut être utilisé dans une variété de contextes et peut être adapté à une variété de types d'utilisation [42]. Il a également une communauté très active. Pour cela on a choisi le langage python pour notre solution.

2.7.1.2 DEAP

Distributed Evolutionary Algorithms in Python (DEAP), est un Framework open source de calcul évolutionnaire pour le prototypage rapide et le test d'idées et des conceptes, développé principalement au Laboratoire de Vision et Systèmes Informatiques de l'Université Laval, Québec, Canada. Il intègre les structures de données et les outils nécessaires à la mise en œuvre des techniques de calcul évolutionnaire les plus courantes, telles que l'algorithme génétique, la programmation génétique, les stratégies d'évolution, l'optimisation par essaims de particules, l'évolution différentielle, l'algorithme de flux de trafic et d'estimation de la distribution.

2.7.2 Réglage des hyperparamètres de l'AG proposé

Nous allons examiner l'influence des différents paramètres génétiques sur les performances de l'**AG** dans le but de déterminer la meilleure combinaison de paramètres. Afin d'évaluer les performances de l'**AG** ainsi que l'impact des paramètres et du choix des méthodes d'évolution sur la qualité des résultats, nous allons procéder à plusieurs simulations avec une variété de cas d'étude. D'abord nous commençons par tester les différentes méthodes d'évolution (mutation et croisement) pour trouver celles qui donnent les meilleures solutions. Ensuite, nous allons étudier l'influence des paramètres voir les probabilités de mutation et de croisement, taille de la population, le nombre de génération ainsi que les dimensions de l'espace de déploiement sur le temps d'exécution et la qualité des solutions obtenues.

2.7.2.1 Impact des opérateurs d'évolutions

L'espace de déploiement étudié et les paramètres d'entrées sont les mêmes pour les différentes simulations. Nous avons pris le cas d'étude qui s'agit d'un espace de $100m^2$ avec 10 m de longueur et 10 m de largeur. Nous exécutons l'algorithme pendant 100 générations et pour une taille de population fixée à 100.

Opérateur de Mutation Nous allons faire une comparaison directe entre les deux méthodes de mutation expliquées précédemment : "Flip Bit" et "Shuffle Indexes". La comparaison est basée principalement sur le temps d'exécution et la qualité de la solution finale.

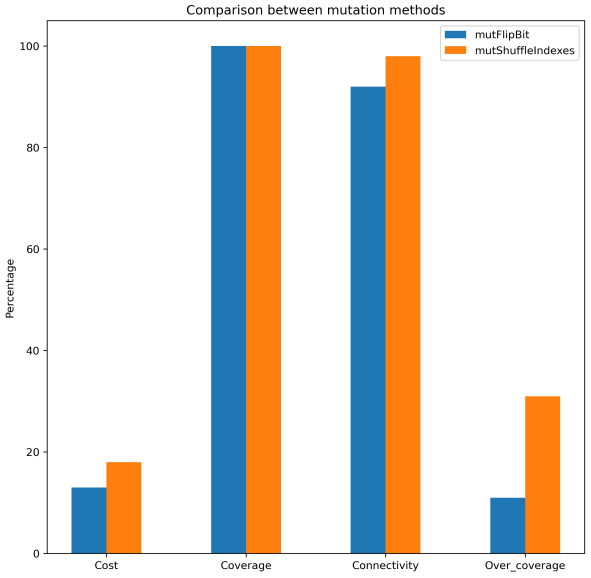

FIGURE 2.13: Comparaison de solutions trouvées par les deux méthodes de mutations.

La figure 2.13 montre clairement que la connectivité et la couverture sont maximisées de manière optimale. Néanmoins, la méthode "Flip Bit" minimise efficacement à la fois le coût et la sur-couverture, alors que l'autre méthode "Shuffle Indexes" ne fournit pas de résultats satisfaisants dans ce cas d'étude ou les dim batiment .

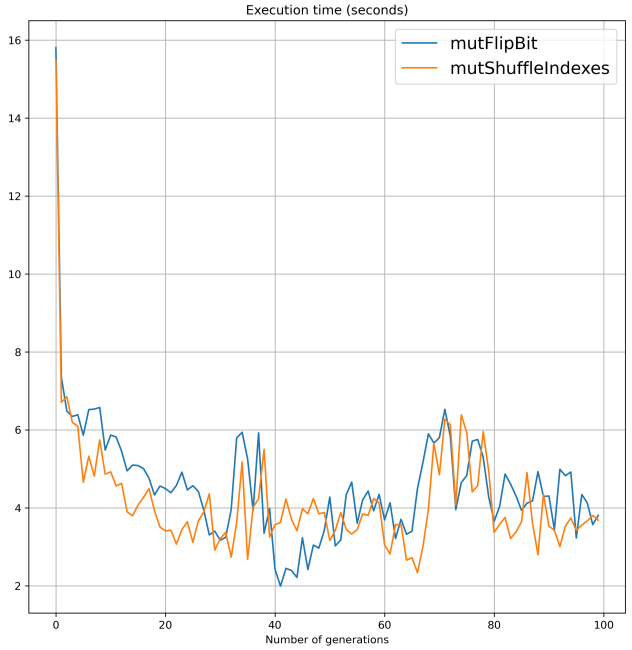

FIGURE 2.14: Comparaison de temps d'exécution des deux méthodes de mutations.

Méthode	Temps d'éxécution (s)	Cout (%)	Connectivité (%)	Couverture (%)	Sur-couverture (%)	Score (%)
mutFlipBit	457.38	13.0	92.0	100.0	11.0	42.0
mutShuffleIndexes	415.15	18.0	98.0	100.0	31.0	37.25

TABLE 2.1: Comparaison entre les deux méthodes de mutation.

Le tableau 2.1 illustre les résultats de ce test, nous pouvons voir que la méthode "Flip Bit" donne des meilleurs résultats que "Shuffle Indexes" en terme de scores des objectifs et donc qualité de solutions, malgré que cette dernière est plus rapide mais la différence en temps d'exécution n'est pas importante comme illustré dans la figure 2.14. Nous avons jugé donc que la méthode "Flip Bit" est plus adéquate pour notre cas.

Opérateur de Croisement Similaire à l'approche précédente, nous allons procéder à une comparaison cette fois-ci entre les deux méthodes de croisement : One Point (croisement à un point), Two Point (croisement à deux points).

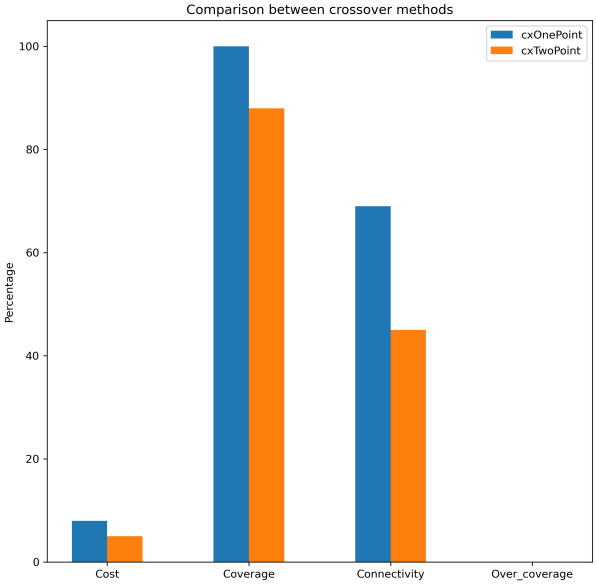

FIGURE 2.15: Comparaison basée sur la qualité de la solution.

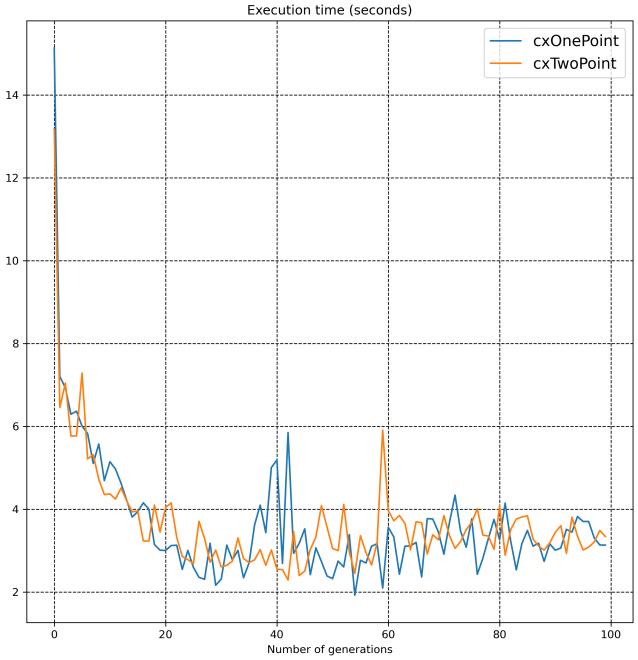

FIGURE 2.16: Comparaison basée sur le temps d'exécution.

Les résultats sont résumés sur le tableau suivant :

Méthode	Temps d'éxécution (s)	Cout (%)	Connectivité (%)	Couverture (%)	Sur-couverture (%)	Score (%)
cxOnePoint	362.03	8.0	69.0	100.0	0.0	40.25
cxTwoPoint	365.69	5.0	45.0	88.0	0.0	32.0

TABLE 2.2: Comparaison entre deux méthodes de croisement.

Nous pouvons constater dans cette simulation ilustré par les Figures 2.15 et 2.16 ainsi que le tableau 2.2 que la surcouverture est entièrement minimisée pour les deux méthodes de croisement utilisées. Nous remarquons aussi que la méthode "Two Point" est la moins bonne que ce soit en temps d'exécution ou la qualité des solutions.

2.7.2.2 Probabilité de mutation et de croisement

Dans un premier temps, nous avons étudié l'influence de la probabilité de mutation et de la probabilité de croisement sur la qualité de la solution et le temps d'exécution. Pour choisir les probabilités les plus adaptées, nous allons varier la probabilité de mutation de 0.01 à 0.1 avec un pas de 0.01. Similairement, nous faisons varier la probabilité de croisement de 0.1 à 0.9. Pour chaque couple, nous enregistrons la valeur de fitness (score) et le temps d'exécution. Les deux tableaux ainsi que les figures représentent les résultats que nous avons trouvés.

Mut/Cro	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.01	314.51	342.31	458.46	343.74	414.73	379.65	475.9	466.59	449.34
0.02	298.36	379.45	477.49	247.78	400.49	409.96	438.03	493.85	389.27
0.03	276.61	344.36	370.98	433.61	341.6	442.08	508.92	524.71	365.85
0.04	273.73	413.62	314.24	400.14	511.33	471.88	515.49	401.8	446.24
0.05	255.09	377.84	364.13	360.7	374.52	362.25	527.07	375.67	445.93
0.06	238.62	324.57	274.19	373.63	382.33	369.16	460.28	399.16	385.33
0.07	255.25	394.7	343.11	362.87	432.99	313.27	439.18	403.55	360.23
0.08	249.0	363.63	351.68	379.29	394.15	361.18	403.56	356.8	469.46
0.09	314.18	380.17	335.31	328.27	342.13	347.79	394.32	343.16	337.7

TABLE 2.3: Temps d'exécution (en secondes) en fonction de la probabilité de mutation et de la probabilité de croisement.

Mut/Cro	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.01	26.0	28.5	29.75	33.25	31.25	33.25	34.75	41.0	35.75
0.02	30.25	31.0	29.75	42.0	34.0	36.75	36.0	35.25	39.75
0.03	31.5	34.75	29.5	28.5	38.0	34.0	37.5	36.0	39.0
0.04	32.25	34.25	35.5	33.25	33.25	34.75	37.75	37.0	36.5
0.05	31.5	34.25	35.0	40.75	34.5	39.25	41.75	38.0	38.5
0.06	34.5	33.0	38.0	38.5	35.25	35.5	39.25	35.75	36.75
0.07	34.25	31.75	34.0	33.25	36.5	38.0	37.5	37.75	39.25
0.08	35.5	32.75	37.0	35.75	36.5	38.25	34.0	40.5	34.75
0.09	32.0	33.75	37.5	36.5	33.75	35.25	38.75	37.25	39.25

TABLE 2.4: Valeur de fitness (score) en fonction de la probabilité de mutation et de la probabilité de croisement.

En fait, on arrive à remarquer que le choix des probabilités de croisement et de mutation affecte de manière critique les performances et le comportement de l'algorithme. Les deux tableaux ainsi que les figures représentent les résultats que nous avons trouvés.

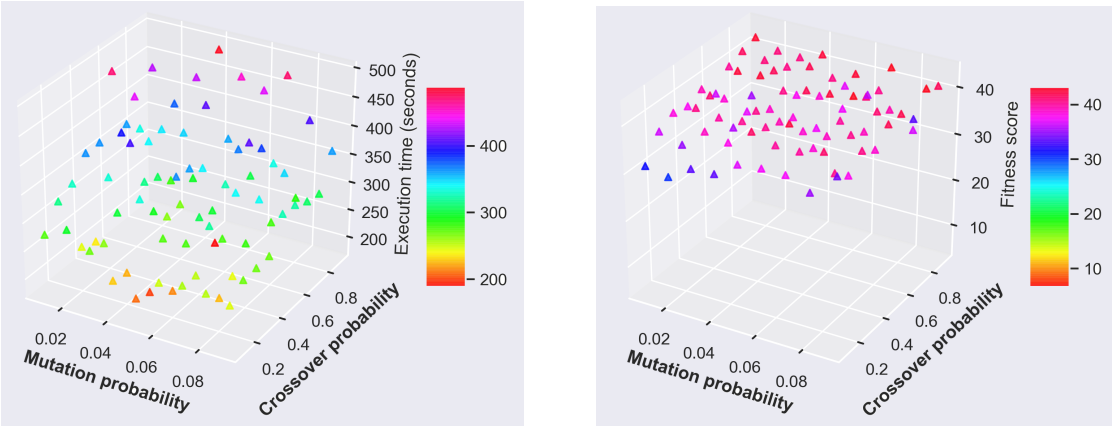

FIGURE 2.17: Variation du temps d'exécution et du score en fonction des probabilité de mutation et de croisement.

Les résultats obtenus ont montré que le temps d'exécution et la qualité de la solution sont proportionnels aux probabilités de croisement et de mutation. De ce fait, plus les deux probabilités de croisement et de mutation sont importantes, plus le processus d'évolution est long. Comme pour les solutions, nous avons constaté que plus la probabilité de croisement est élevée, plus les solutions données par les algorithmes sont performantes et plus elles

satisfont nos quatre objectifs. Nous avons donc défini la probabilité de croisement à 0,9 et la probabilité de mutation à 0,1, pour le reste des expérimentations.

2.7.2.3 Nombre de générations

En effet, il est évident que si le nombre de générations est fixé à un niveau élevé, le temps d'exécution de notre algorithme sera plus long comme nous pouvons le voir dans le tableau 2.5 ci-dessous.

Nombre de générations	Temps d'exécution (s)	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
20	100.35	26	99	100	53	30
40	164.47	15	91	100	27	37
60	233.86	18	99	100	37	36
80	330.62	16	95	100	24	38.75
100	419.35	17	98	100	26	38.75

TABLE 2.5: Influence du nombre de générations sur la qualité de la solution et le temps d'exécution (en seconde).

Pour un petit nombre de générations, le temps d'exécution est assez faible au détriment de la qualité de la solution. Nous pouvons remarquer que la solution s'améliore de plus en plus en fonction du nombre de générations car ce dernier garantit que l'algorithme découvre de nouveaux individus dans l'espace de recherche, ce qui apporte de la diversité dans sa population et donc des solutions de meilleure qualité.

2.7.2.4 Taille de la population

Pour commencer, nous avons fait varier la taille de la population initiale de 20 jusqu'à 100 individus avec un pas de 20 et pour un nombre de générations fixé à 100. Nous observons sur la figure 2.18 que le temps d'exécution est proportionnel à la taille de la population.

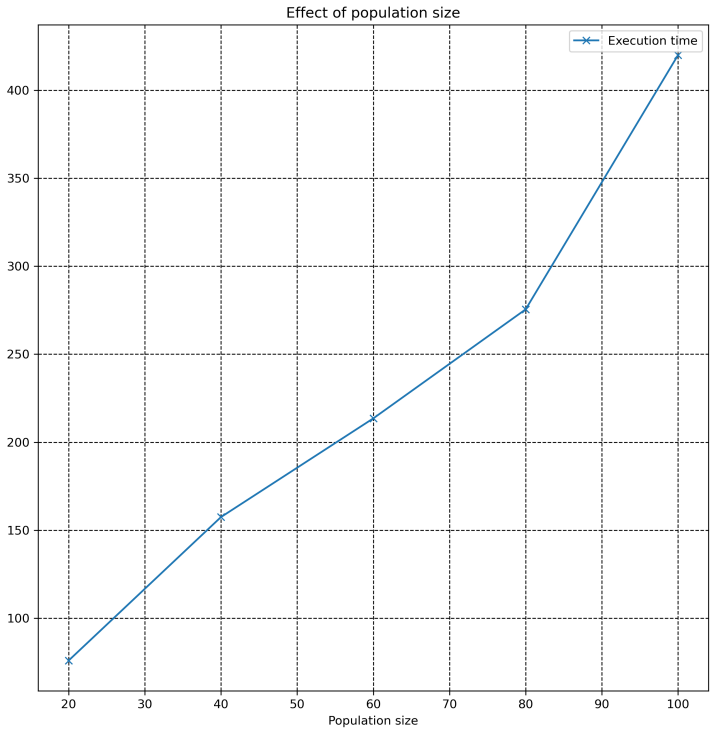

FIGURE 2.18: Influence de la taille de population sur le temps d'exécution (en seconde).

La figure 2.19 montre clairement que pour une taille de population plus élevée, les lignes de temps d'exécution sont superposées, ce qui montre clairement l'effet sur le temps d'exécution dont nous avons parlé précédemment.

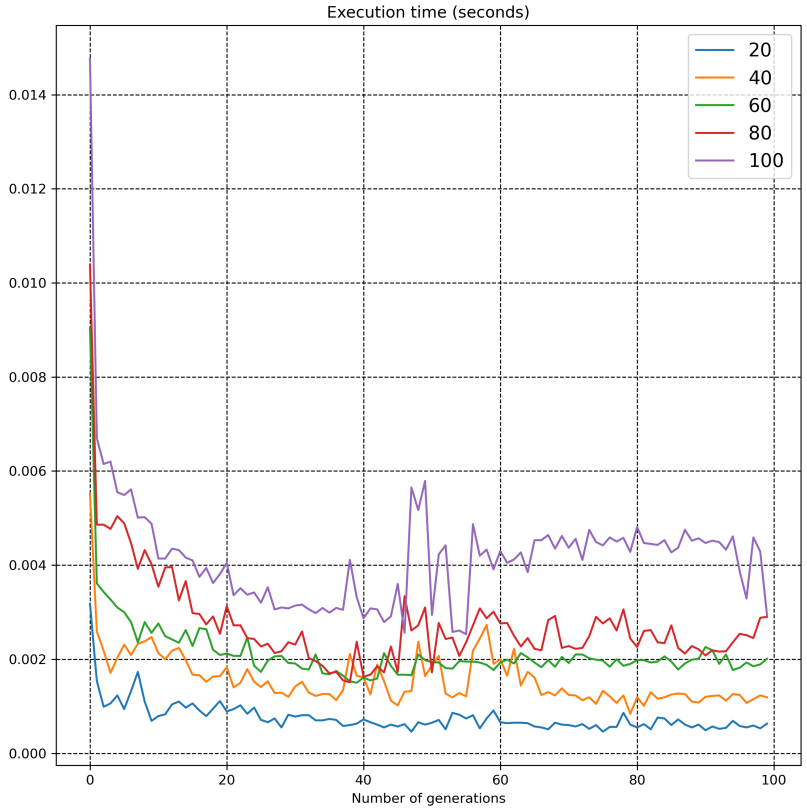

FIGURE 2.19: Temps d'exécution (en seconde) en fonction de générations pour differentes tailles de population.

La solution finale pour chaque exécution est représentée dans le tableau 2.6. Nous avons constaté que si la taille de la population est petite, les solutions trouvées ne répondent pas à nos besoins. Cela est dû au fait que les algorithmes ne peuvent pas explorer complètement l'espace de recherche avec une petite population, et qu'ils se retrouvent donc dans des optimums locaux.

Taille de population	Temps d'exécution (s)	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
20	75.92	19.0	90.0	99.0	38.0	33.0
40	157.45	17.0	88.0	100.0	30.0	35.25
60	213.53	18.0	89.0	100.0	33.0	34.5
80	**275.48**	**13.0**	**91.0**	**100.0**	**6.0**	**43.0**
100	419.94	17.0	98.0	100.0	26.0	38.75

TABLE 2.6: Influence de la taille de population sur la qualité de la solution et le temps d'exécution (en seconde).

Pour remédier à ce problème, nous avons décidé de fixer la taille de la population à 80 individus. Nous avons constaté que les solutions obtenues avec cette taille sont satisfaisantes. De plus, le temps d'exécution n'est pas aussi important (285.48 secondes).

2.7.3 Choix de l'AGMO (SPEA-II vs NSGA-II)

Cette fois, nous allons fixer le nombre de générations à 100, tandis que la taille de la population est fixée à 80. Les deux algorithmes multi-objectifs discutés précédemment ont été comparés l'un à l'autre.

La figure 2.20 met clairement en évidence la différence de qualité pour chaque objectif. Il en ressort que les solutions non-dominées obtenues par SPEA-II sont meilleures que celles de NSGA-II en termes de convergence et de diversité. Bien que SPEA-II donne de meilleurs résultats, NSGA-II fournit toujours des résultats acceptables, et il parvient tout de même à être efficace.

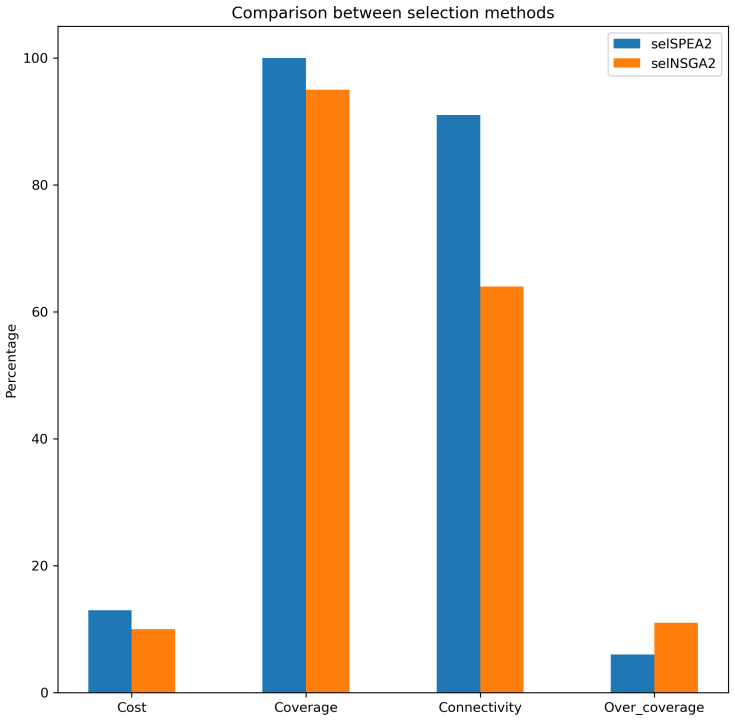

FIGURE 2.20: Valeurs de fitness des 4 objectifs (coût, couverture, connectivité et surcouverture) en utilisant NSGA et SPEA.

Néanmoins, il a été observé sur la figure 2.21 que pour ce problème particulier, au fur et à mesure que la simulation progresse vers les meilleures solutions, SPEA-II prend un peu plus de temps d'exécution par rapport à NSGA-II.

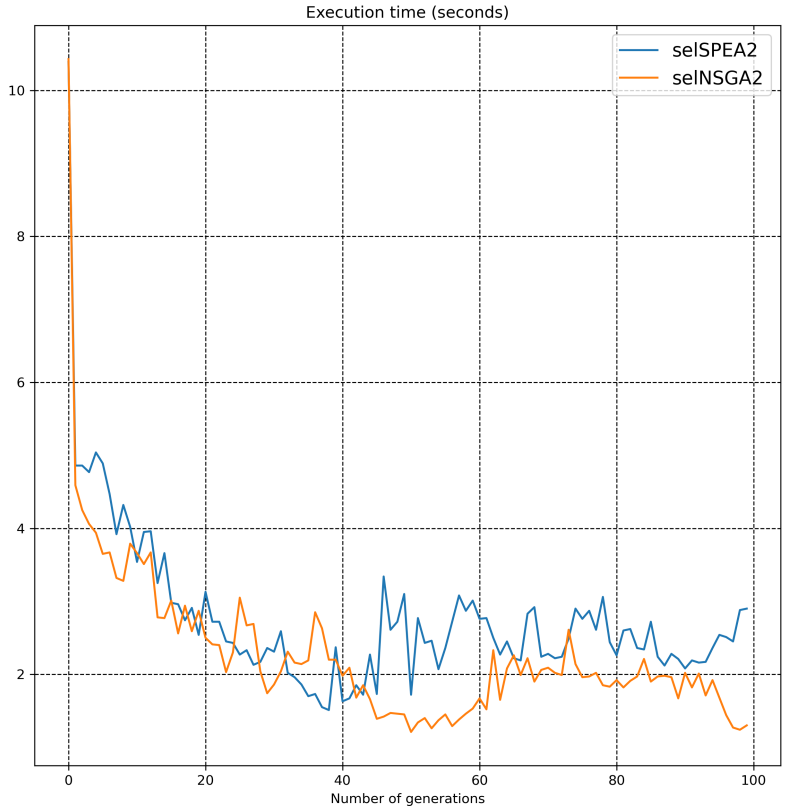

FIGURE 2.21: Evolution du temps d'exécution en fonction de générations.

Les résultats de cette simulation sont présentés dans le tableau suivant 2.7 :

Algorithme	Temps d'exécution (s)	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture	Score (%)
SPEA2	275.48	13.0	91.0	100.0	6.0	43.0
NSGA2	227.77	10.0	64.0	95.0	11.0	34.5

TABLE 2.7: Influence du choix de l'algorithme multi-objectif sur la qualité de la solution et le temps d'exécution (en seconde).

2.7.4 Exemples de déploiement d'un RCSF

Afin de mettre à l'épreuve et les performances de l'AGMO, nous avons pris un cas d'étude. Il s'agit d'un espace de $100m^2$: 10m de longueur et 10m de largueur. Nous allons utiliser l'AGMO SPEA-II qui est jugé plus performant dans les simulations précédentes.

La figure 2.22 illustre parfaitement la qualité de la solution en ce qui concerne la bonne optimisation, c'est à dire le coût et la sur-couverture sont minimisés avec efficacité, alors que la connectivité et la couverture sont maximisées à des valeurs très satisfaisantes au-dessus de 90%.

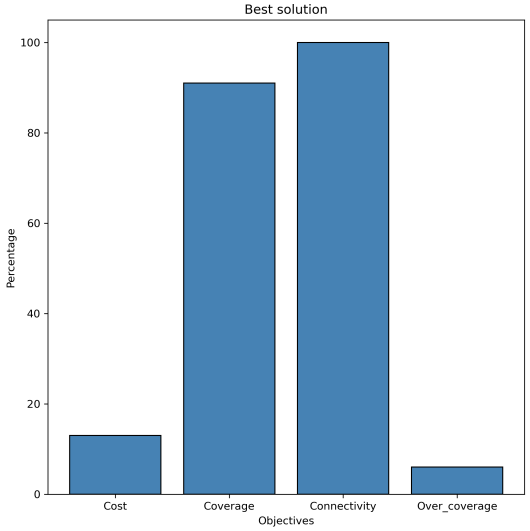

FIGURE 2.22: Valeurs de fitness représentées dans un diagramme à barres.

Afin de s'assurer que notre algorithme (SPEA-II) converge vers une solution appropriée, nous comparons l'évolution de la valeur de fitness de la solution à l'ensemble de la population. Cela nous donnera un aperçu de la façon dont l'optimisation de l'objectif se produit dans son espace de recherche correspondant. Cela est possible en calculant une valeur d'aptitude moyenne pour l'ensemble de la population et en la comparant à la valeur de la solution. La figure 2.23 montre le comportement de chaque objectif lors de l'exécution de l'algorithme pendant 100 générations.

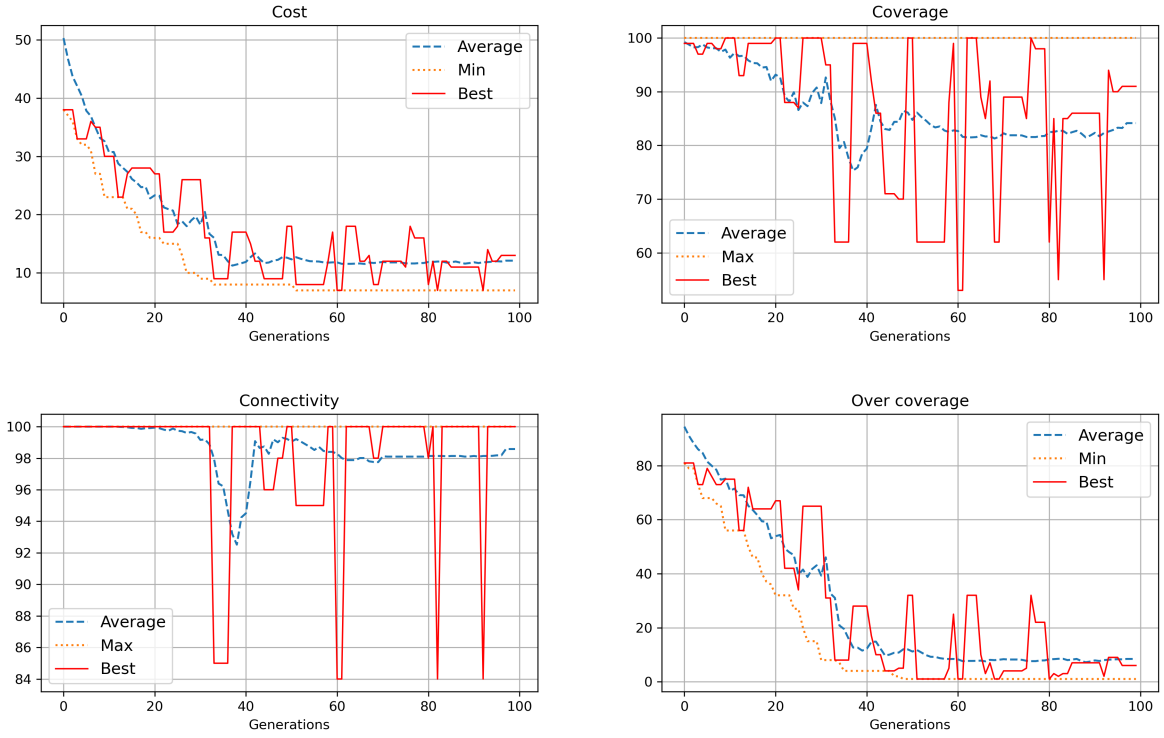

FIGURE 2.23: Évolution des valeurs de fitness pour les 4 objectifs.

Par exemple, le coût, qui est un critère qui doit être minimisé, son comportement est observé en comparant la valeur de fitness de la meilleure solution (en rouge) pour chaque génération à celle de la valeur moyenne (en bleu), c'est à dire à celle de l'espace de recherche. Comme on peut le voir, les deux courbes rouge et bleue sont pratiquement proches l'une de l'autre, ce qui garantit une bonne minimisation. C'est important car cela donne une idée de la qualité des solutions au niveau de la population. En revanche, pour un problème de maximisation, nous cherchons à ce que la valeur de fitness moyenne soit la plus élevée possible.

En général, nous avons également tendance à comparer la valeur de fitness de notre solution (en rouge) à la valeur de fitness minimale (en jaune) qui existe dans l'espace de recherche. Dans ce cas, nous voulons que la valeur d'aptitude de notre solution converge vers la valeur minimale.

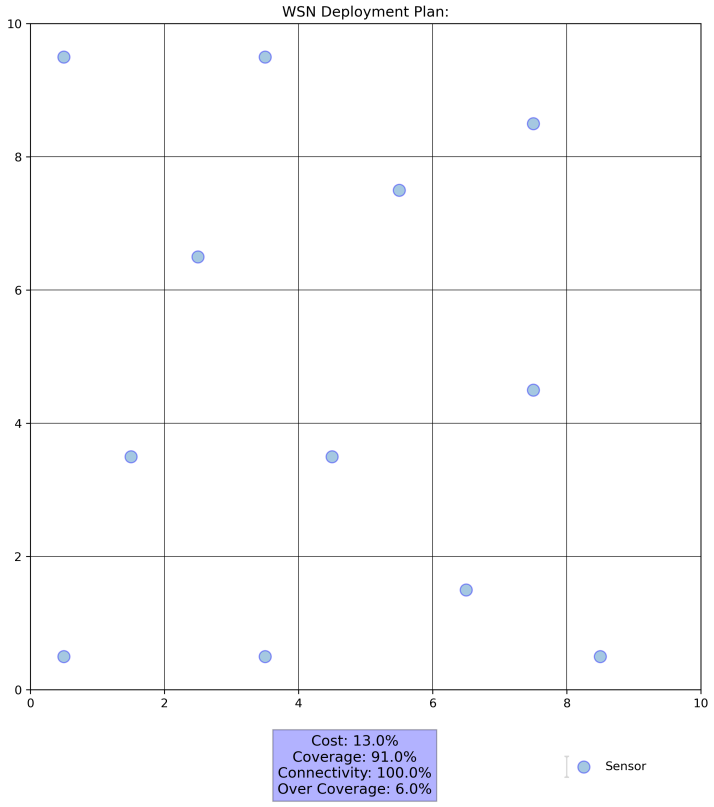

FIGURE 2.24: Plan de déploiement de la solution.

La figure ci-dessus 2.24 montre un exemple de plan de déploiement pour cette solution, où chaque nœud représente un placement de capteur.

Solutions	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
1	13.0	91.0	100.0	6.0	43.0
2	10.0	80.0	100.0	4.0	41.5
3	15.0	95.0	100.0	12.0	42.0
4	13.0	91.0	100.0	6.0	43.0
5	8.0	62.0	98.0	1.0	37.75
6	9.0	73.0	99.0	2.0	40.25
7	16.0	97.0	100.0	17.0	41.0
8	12.0	89.0	100.0	4.0	43.25
9	11.0	86.0	100.0	7.0	42.0
10	17.0	99.0	100.0	24.0	39.5

TABLE 2.8: Exemple de solutions données par l'algorithme SPEA-II.

Nous lançons l'algorithme SPEA-II et nous prenons note des solutions obtenues à la fin de l'exécution. Une partie des solutions générées est ainsi représentée dans le tableau 2.8. Il revient donc à l'utilisateur de sélectionner la solution qui lui semble la plus adaptée en fonction de l'ordre d'importance qu'il accorde aux critères de déploiement.

2.8 Analyse des performances

Dans cette partie du chapitre, nous allons voir l'impact sur la consommation de temps par les différentes phases de l'AG et la qualité des solution de ce dernier. Cet impact peut généralement être causée par différents paramètres comme nous l'avons vu dans les simulations précédentes, comme l'impact de la taille de la population et le nombre de générations. Cependant, il existe un autre facteur qui est très important et qui est lié au problème du déploiement. Il s'agit des dimensions du bâtiment dans lequel nous implémentons le RCSF.

2.8.1 Résultats

Dans les simulations ci-dessous, nous avons exécuté SPEA-II pour différentes tailles de bâtiments. Nous avons également veillé à conserver les mêmes entrées de paramètres génétiques comme constantes. Nous l'avons initialisé avec une population fixée à 80 individus et un nombre maximum de générations établi à 100. Quant aux autres paramètres, nous avons gardé les mêmes que ceux que nous avons discutés précédemment.

2.8.1.1 Impact des dimensions sur la qualité

Dans un premier temps, nous allons jeter un coup d'œil sur la qualité des solutions illustré dans le tableau 2.9. Nous remarquons que la qualité de notre algorithme commence à diminuer lorsque les dimensions du bâtiment commencent à augmenter.

Dimensions	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
$64m^2$	9.38	78.12	98.44	1.56	41.41
$100m^2$	17.0	98.0	100.0	26.0	38.75
$225m^2$	23.56	100.0	100.0	53.33	30.78
$400m^2$	25.25	97.0	100.0	60.0	27.94

TABLE 2.9: Influence des dimenions du bâtiment sur la qualité de la solution.

2.8.1.2 Temps consommés

Cette perte de qualité s'accompagne également d'une augmentation drastique du temps de calcul, comme le montre le tableau ci-dessous 2.10. La consommation de temps par les phases de la génétique est exploitée en détail.

Dimensions	Temps d'exécution (s)	Sélection (s)	Croisement (s)	Mutation (s)	Evaluation (s)
$64m^2$	298.29	8.97	0.0	0.0	98.66
$100m^2$	419.94	9.75	0.04	0.01	253.94
$225m^2$	1576.78	12.65	0.02	0.06	1454.85
$400m^2$	6038.13	17.74	0.41	0.16	5852.39

TABLE 2.10: Influence des dimenions du bâtiment sur le temps d'exécution (en seconde).

Les phases de mutation et de croisement semblent avoir un impact quasi nul sur le temps d'exécution global. De même, le processus de sélection peut prendre quelques secondes mais il est négligeable par rapport à la phase d'évaluation qui est très exigeante en termes de dépenses de calcul.

Pour un bâtiment de 20 m de long et 20 m de large, le temps d'exécution total peut devenir très fastidieux et atteindre plus de 6000 secondes, soit presque plus d'une heure et demie. La figure 2.25, montre l'évolution de la consommation de temps par les différentes phases de l'algorithme génétique sur 100 générations. On observe que la courbe du temps d'exécution de la phase d'évaluation correspond au temps d'exécution total. Cependant, à la première génération, il y a une différence de 110.53 secondes entre les deux courbes, qui est due à la phase d'initialisation où nous générons la première population aléatoirement et évaluons chacun de ses individus. Quant aux autres phases, elles sont clairement négligeables.

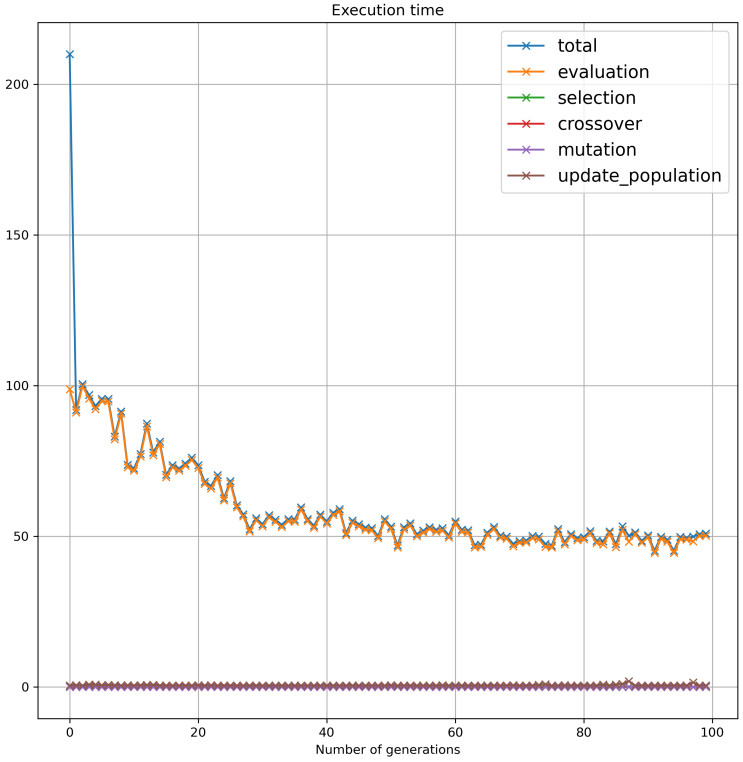

FIGURE 2.25: Evolution du temps d'exécution en fonction de générations pour un bâtiment 20x20.

Cette augmentation du temps d'exécution est proportionnelle au nombre d'entrées qui sont définies pour l'algorithme. L'entrée n'est rien d'autre que la taille d'un individu, c'est à dire la longueur du vecteur binaire qui est défini comme le chromosome représentant la solution.

2.8.2 Discussion

Les résultats précédents peuvent être expliqués par le fait que le déploiement d'un RCSF peut être considéré comme NP-hard. Un problème est dit NP-hard si il prend peu de temps pour vérifier la qualité de la solution mais il demande beaucoup d'efforts et de temps d'exécution pour être résolu. Par exemple, un problème de sudoku [43], est considéré comme NP-hard parce qu'il est difficile à résoudre car il y a beaucoup de solutions possibles, mais une fois qu'il est résolu, nous pouvons vérifier la solution facilement. Dans notre cas, nous pouvons vérifier si notre solution est valide en évaluant facilement les 4 objectifs. La vérification ne prend pas beaucoup de temps. D'un autre côté, trouver la solution est très difficile et

prend beaucoup de temps à calculer, car cela peut devenir très coûteux en termes de calcul et le temps d'exécution croît exponentiellement.

2.9 Conclusion

Dans ce chapitre, nous avons présenté les **AG**s et leur analogie avec l'évolution Darwinienne, puis nous avons détaillé ses principes de fonctionnement de base et l'avons également adapté au problème du déploiement du RCSF. Nous avons également modélisé ce problème mathématiquement et introduit le concept de fonction de fitness pour comparer la qualité des solutions possibles concernant les critères de déploiement tels que le coût, la connectivité, la couverture et la sur-couverture. Nous avons pu sélectionner l'algorithme multi-objectif approprié (SPEA-II) et définir la meilleure combinaison de paramètres génétiques à utiliser pour notre cas d'étude.

Nous avons confirmé que les AGs peuvent résoudre des problèmes multiobjectifs complexes en peu de temps et offrir des solutions exploitables. Cependant, leur inconvénient est qu'ils ont plusieurs paramètres qui doivent être réglés en conséquence. De plus, la phase d'évaluation semble avoir un impact énorme sur l'efficacité de l'algorithme lui-même, qui est principalement liée au problème lui-même et non à l'algorithme explicitement.

Nous concluons que l'AG peut être encore amélioré en trouvant un moyen de réduire le temps d'évaluation de la valeur de fitness et donc de le rendre moins gourmand en calcul et améliorer la qualité des solutions dans le cas des grands bâtiments, en s'appuyant sur des métaheurstiques hybrides qui seront explorées dans le prochain chapitre.

Chapitre 3

Métaheuristiques hybrides pour le déploiement d'un RCSF

3.1 Introduction

Nous avons vu dans le chapitre précédent une approche évolutionnaire pour résoudre le problème du déploiement d'un RCSF, cela en utilisant des AGs multi-objectifs (NSGA-II et SPEA-II). Les résultats nous dévoilent que ce type d'algorithmes est adéquat pour cette problématique avec le choix des méthodes et paramètres d'évolution, mais les différentes simulations montrent que cette méthode est limitée quand l'espace de déploiement est de grande taille, l'algorithme devient gourmand en temps de calcul et en mémoire ou il a pris plus d'une heure pour s'exécuter, ainsi il a tendance à ne pas converger vers l'optimum global. De plus à traver une analyse des temps consommés par les phase de l'algorithmes dans le chapitre passé, nous constatons que c'est la phase d'évaluation qui prend autant pour s'exécuter.

Pour surmonter ce problème, dans ce chapitre nous allons proposer une amélioration de l'AG utilisée pour réduire le temps d'exécution et maintenir une bonne qualité de solution, pour cela nous allons limiter l'utilisation de l'évaluation directe en la remplaçant par un modèle d'approximation. En effet, plusieurs travaux montrent que l'utilisation de modèles d'approximation dans les algorithmes évolutionnaires est appréciée pour ce genre de problèmes. Dans ce chapitre, nous présentons la notion d'hybridation dans les métaheuristiques, ainsi que les différents modèles approximatifs qui existent dans la littérature, afin de construire notre modèle d'approximation et le tester en comparant ces résultats avec ceux obtenus dans le chapitre précédent.

3.2 Hybridation des métaheuristiques

Dans le domaine de l'optimisation, une hybridation de méthodes consiste à combiner plusieurs (deux ou plus) méthodes afin de tirer les avantages et améliorer les performances de l'algorithme (cette définition ne concerne pas que les algorithmes du domaine d'optimisation). Ces dernières années le domaine d'hybridation de méthodes d'optimisation a connu un très grand succès dans la communauté scientifique (recherche opérationnelle). Selon Gunther R. Raidl [44], les algorithmes hybrides sont classés selon les problématiques traitées : problèmes de conception liés à l'algorithme lui-même, son fonctionnement et l'architecture de l'algorithme hybride lui-même. Il y a aussi les problèmes de mise en œuvre liés au matériel utilisé (hardware), au modèle de programmation et à l'environnement d'exécution de l'algorithme.

3.2.1 Problèmes de conception

Les algorithmes hybrides qui traitent les problèmes de conception sont classés hiérarchiquement : hybride de bas niveau et hybride de haut niveau comme montre la Figure 3.1. Ils peuvent aussi être classés horizontalement, voir Figure 3.2.

3.2.1.1 Classification hiérarchique

Hybridation à bas-niveau La métaheuristique (un réseau de neurones ou une fonction de régression par exemple) va remplacer une fonction quelconque (ex. : mutation, croisement ou évaluation dans notre cas d'étude).

Hybridation à haut-niveau Dans ce type d'hybridation les métaheuristiques sont indépendants. Il n'existe aucune relation directe entre le déroulement et le fonctionnement de ces derniers, les algorithmes combinés sont considérés comme indépendants l'un de l'autre.

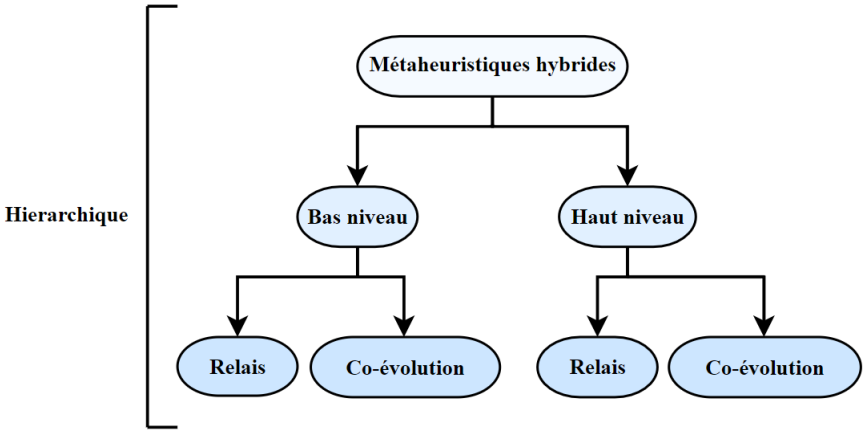

FIGURE 3.1: Schéma de classification hiérarchique des métaheuristiques hybrides.

Pour chacun des deux niveaux, il existe deux manières de fonctionnement :
- Relais : les métaheuristiques sont exécutés de manière séquentielle, ou la solution de départ d'un algorithme dépend de celle de l'algorithme précédent.
- Travail d'équipe (Co-évolution) : l'hybridation est réalisée en parallèle , dans laquelle plusieurs métaheuristiques s'exécutent en même temps. Donc chaque algorithme effectue sa propre recherche dans l'espace de solution.

3.2.1.2 Classification horizontale (Plate)

Pour ce type de classification on a trois familles distinctes d'algorithmes d'hybridation des méta heuristiques qui sont classées comme suit :

Le type d'hybridation (homogène, hétérogène) : l'hybridation homogène est l'utilisation de plusieurs algorithmes du même type, exemple une combinaison de plusieurs algorithmes évolutionnaires . Pour le cas d'hybridation hétérogène, les métaheuristiques utilisées sont différentes. C'est souvent une S-méthauristique telle que la recherche tabou qui va générer des solutions qui seront intégrées dans la population d'un algorithme génétique.

Le domaine de recherche (global, partiel) : Dans une hybridation de domaine partiel, l'espace est décomposé en plusieurs sous-espaces en vue d'affecter chaque sous-espace a une métaheuristique spécifique. Alors que dans une hybridation de domaine globale, on utilise plusieurs métaheuristiques qui explorent le même espace de recherche.

La fonction objective utilisée (générale, spécialiste) : il en existe deux types : générale ou toutes les métaheuristiques ont utilisé la même fonction objective et donc sont utilisées pour résoudre le même problème d'optimisation. Alors que dans le cas spécialisé plusieurs méta heuristiques sont utilisées et chacune résout un problème spécifique exprimé par sa fonction objective.

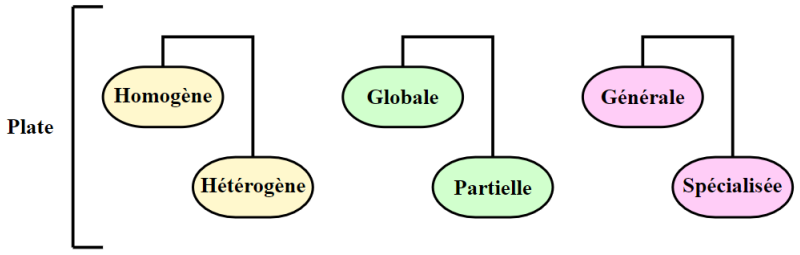

FIGURE 3.2: Schéma de classification horizontale des métaheuristiques hybrides

3.2.2 Problème d'implémentation

Pour ces problèmes, l'environnement dans lequel l'algorithme est exécuté représente le vif point à étudier. Dans [45], les algorithmes sont classés selon le matériel utilisé : ordinateurs à usage général ou on distingue deux catégories selon la manière avec laquelle ils sont exécutés parallèle ou séquentielle. Ainsi que les ordinateurs spécifiques ou les algorithmes sont classées selon le hardware utilisé : GPU, CPU, FPGA.

Dans notre cas d'étude, le problème que nous allons traiter est un problème de conception, nous nous intéressons aux architectures hybrides bas-niveau travail-d'équipe (Low level Teamwork Hybrid), donc l'algorithme hybride développé consiste de l'algorithme génétique

utilisé dans le deuxième chapitre et une métaheuristique qui va remplacer la fonction d'évaluation qui consomme trop de ressources et temps de calcule sur tout quand il s'agit des problèmes de grande dimensions, ou le temps d'exécution de l'AG augmente exponentiellement en fonction des dimensions de la surface du déploiement.

3.3 Approximation de la valeur de fitness

Comme mentionné précédemment l'évaluation des valeurs de fitness peut être très coûteuse en termes de calculs, Il existe aussi plusieurs situations dans lesquelles cette opération devient très difficile ou même impossible à exprimer analytiquement. Il est donc envisageable d'avoir une approximation efficace en minimisant le nombre d'évaluations de fitness afin de trouver une solution acceptable. Pour pallier ce problème, des modèles efficaces en termes de calcul peuvent être construits pour approximer la fonction objective. Ces modèles sont souvent appelés modèles approximatifs ou méta-modèles. L'approximation de la valeur de fitness est généralement utilisée dans les cas suivants :

— Temps de calcul énorme.

— Modèle d'évaluation inexistant ou trop complexe.

L'idéal serait qu'un modèle approximatif puisse remplacer entièrement la fonction objective originale. Cependant, selon l'article [46], les chercheurs ont réalisé qu'il est en général nécessaire de combiner le modèle approximatif avec la fonction objective originale pour que l'AG converge correctement. À cette fin, la réévaluation de certains individus à l'aide de la fonction d'adéquation originale, également appelée contrôle de l'évolution, est essentielle.
En raison du manque de données pour l'apprentissage des modèles approximatifs, ainsi que la haute dimensionnalité de l'espace d'entrée, il est très difficile d'obtenir une approximation fonctionnelle globale parfaite de la fonction d'évaluation originale. Pour résoudre ce problème, deux approches principales peuvent être suivies :

— Premièrement, le modèle approximatif doit être utilisé avec la fonction d'évaluation originale. Dans la plupart des cas, elle est disponible, bien qu'elle soit très coûteuse en termes de calcul. Il est donc très important de l'utiliser efficacement. C'est ce qu'on appelle le contrôle de l'évolution dans le calcul évolutionnaire.

— Deuxièmement, la qualité du modèle approximatif doit être améliorée autant que possible compte tenu d'un nombre limité de données. Plusieurs aspects sont importants pour améliorer la qualité du modèle, tels que la sélection du modèle, la sélection de la méthode d'apprentissage et la sélection des mesures d'erreur.

3.3.1 Types d'approximations

Le concept d'approximation en optimisation n'est pas nouveau. Traditionnellement, il existe deux approches de base, à savoir l'approximation fonctionnelle et l'approximation du problème. En outre, des techniques d'approximation spéciales pour l'évaluation de fitness ont également été proposées [46].

3.3.1.1 Approximation du problème

Dans cette approche, l'énoncé du problème original lui-même est remplacé par un problème réduit, autre qui est approximativement plus facile à résoudre.

3.3.1.2 Approximation fonctionnelle

Dans l'approximation fonctionnelle, le calcul est fait grâce à un modèle mathématique simple. Quand la fonction fitness devient très gourmande en temps de calcul, elle peut être remplacée par une expression plus simple où les entrées sont les individus à évaluer et ses sorties sont les valeurs de la fonction fitness.

3.3.1.3 Approximation évolutionnelle

Ce type d'approximation est spécifique aux algorithmes évolutionnaires. Elle se divise en deux classes d'algorithmes :

Héritage de la forme physique La fonction fitness des individus issus de la phase de reproduction est héritée à partir de la valeur de fitness de leurs parents. Par exemple, on peut estimer la valeur de fitness d'un enfant par une somme pondérée.

Imitation de la forme physique Les individus sont regroupés en plusieurs groupes grâce à une méthode de classification. Ensuite, seul l'individu représentant son groupe sera évalué à l'aide de la fonction de fitness. Tandis que les valeurs des individus du même groupe seront estimées à partir de l'individu représentatif sur la base d'une mesure de distance.

Il existe plusieurs modèles d'approximation tels que : réseaux de neurones, régression et interpolation, modèle de Krigeage.

3.3.2 Incorporation de modèles de fitness approximatifs

L'utilisation de modèles d'approximation pour les évaluations de fitness peut réduire le nombre d'évaluations de fitness de la manière la plus significative [47]. Cependant, l'application des modèles d'approximation au calcul évolutionnaire n'est pas aussi simple qu'on pourrait le croire. Un point essentiel est qu'il est très difficile de construire un modèle approximatif qui soit globalement correct en raison du nombre limité d'échantillons d'apprentissage.
On constate que si un modèle approximatif est utilisé pour l'évaluation de la fitness, il est très probable que l'AG converge vers un faux optimum. Un faux optimum est un optimum du modèle approximatif, qui n'est pas celui de la fonction de fitness originale comme montre la Figure 3.3.

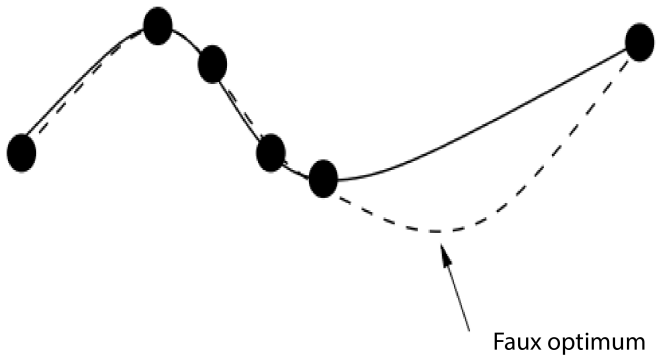

FIGURE 3.3: Faux Optimum d'un modèle approximatif

Par conséquent, il est essentiel, dans la plupart des cas, que le modèle approximatif soit utilisé avec la fonction de fitness originale. On dit que l'évolution est contrôlée.

3.3.2.1 Contrôle d'évolution

Dans le calcul évolutionnaire utilisant des modèles approximatifs, la fonction de fitness originale est utilisée pour évaluer certains des individus ou tous les individus dans certaines générations, cette approche est appelée contrôle d'évolution dans [48].
L'importance d'utiliser à la fois le modèle approximatif et la fonction d'origine pour l'évaluation de l'adéquation a été reconnue. Il existe généralement deux types de contrôle de l'évolution, l'un basé sur l'individu, l'autre sur la génération :

Contrôle basé sur l'individu A chaque génération, certains individus utilisent le modèle approximatif pour l'évaluation de fitness et d'autres la fonction originale. Un individu qui est évalué à l'aide de la fonction de fitness originale est appelé un individu contrôlé.

Contrôle basé sur les générations Dans ce cas, une génération dans laquelle tous les individus sont évalués à l'aide de la fonction fitness originale est appelée une génération contrôlée. Le contrôle est effectué une fois par un nombre fixe de générations et les autres générations sont évaluées grâce à la fonction approximative.

3.3.3 Apprentissage automatique supervisé

Le terme **"apprentissage automatique"** fait généralement référence à un programme informatique qui reçoit des entrées et produit des sorties [49]. L'objectif est d'entraîner ce programme, également appelé modèle, pour produire des sorties correctes qui correspondent aux entrées données, sans les programmer explicitement.

Au cours de ce processus d'apprentissage, le modèle apprend la relation entre les entrées et les sorties en ajustant ses paramètres internes. Une façon courante d'entraîner le modèle consiste à lui fournir un ensemble d'entrées, pour lesquelles la sortie correcte est connue. Pour chacune de ces entrées, nous indiquons au modèle quelle est la sortie correcte afin qu'il puisse s'ajuster, ou se régler, dans le but de produire finalement la sortie souhaitée pour chacune des entrées données. Ce réglage est au cœur du processus d'apprentissage.

Les deux principaux types d'apprentissage automatique supervisé sont la classification et la régression, et seront décrits dans les sous-sections suivantes. Et comme nous l'avons mentionné précédemment, chaque modèle d'apprentissage supervisé est constitué d'un ensemble de paramètres internes réglables et d'un algorithme qui règle ces paramètres pour tenter d'obtenir le résultat souhaité.

3.3.4 Modèles d'approximation

Une étude comparative entre les différentes méthodes d'approximation dans [46], montre que l'utilisation de ces méthodes est efficace pour résoudre des problèmes d'optimisation multi-objectifs en peu de temps. Une classification des méthodes d'approximation, selon l'approche utilisée, a été proposée par l'auteurs, comme suit : Instance Based Learning,

Machine Learning et Statistical Learning. Le choix des méthodes de classifications et d'approximation dépend des besoins de l'utilisateur et de la complexité du problème tels que :

3.3.4.1 Régression linéaire

Le modèle d'approximation polynomiale le plus largement utilisé est le modèle du second ordre [50]. La méthode des moindres carrés (LSM) peut être utilisée pour estimer les coefficients inconnus du modèle polynomial mais le principal inconvénient est que le coût de calcul devient inacceptable lorsque la dimensionnalité augmente. Pour résoudre ce problème, la méthode du gradient peut être utilisée car elle est beaucoup plus simple et n'a pas besoin de calculer la dérivée seconde. Par contre, cette méthode peut être très lente et le nombre d'itérations dépend largement de l'échelle du problème.

3.3.4.2 Réseaux de neurones

Les réseaux de neurones (**RNs**) sont la technique la plus utilisée pour la classification, l'estimation, la prédiction et la segmentation [51]. Ils sont issus de modèles d'inspiration biologique et sont constitués d'unités de base (neurones) organisées selon une architecture spécifique. Ils sont généralement optimisés grâce à des méthodes d'apprentissage statistique, afin qu'ils puissent prendre des décisions basées sur la perception plutôt que sur un raisonnement logique formel. Un réseau de neurones artificiels se compose de plusieurs neurones, qui sont connectés par des liens afin qu'ils puissent envoyer et recevoir des signaux des neurones qui les précèdent. Chacune de ces connexions reçoit un poids, qui détermine son influence sur le neurone connecté. Par conséquent, chaque neurone a une entrée pour pouvoir recevoir des informations d'autres neurones, il peut également avoir une fonction d'activation, et enfin une sortie.

3.3.4.3 Arbres de décisions

Un arbre de décision (DT) [52] est un diagramme ou un graphique qui aide à déterminer un plan d'action ou à montrer une probabilité statistique. Le diagramme est appelé arbre de décision en raison de sa ressemblance avec la plante éponyme, généralement décrite comme un diagramme vertical ou horizontal qui se ramifie. À partir de la décision elle-même (appelée "node"), chaque "branch" de l'arbre de décision représente une décision, un résultat ou une réaction possible. Les branches les plus éloignées de l'arbre représentent les résultats finaux d'une certaine voie de décision et sont appelées les "feuilles". Les DTs sont souvent

utilisés comme un modèle de régression dans les différents domaines d'optimisation. En mathématiques, les DTs sont aussi appelés diagrammes en arbre.

3.3.5 Travaux récents

Il existe plusieurs travaux qui s'intéressent à ces modèles approximatifs afin d'approcher les valeurs de fitness et de réduire le nombre d'évaluations directes étant donné que le temps de calculs et la consommation des ressources mémoire sont des paramètres qui ont un impact sur la qualité de l'AG.

Pour cela plusieurs méthodes ont été proposées dont la plus simple est une méthode d'estimation qui se base sur la distance entre les individus [53], la population de l'AG est regroupée en un certain nombre de classes et le centroid de chaque classe est considéré comme l'individu représentatif de cette classe est évalué en utilisant la fonction d'évaluation originale. La valeur de fitness des autres individus du même groupe est estimée en fonction de leur distance euclidienne par rapport aux centroïdes. Évidemment, ce type d'estimation est très approximatif et la caractéristique originale de l'aspect fitness et son sens physique sont complètement ignorés.
Pour surmonter ce problème, dans [54] les auteurs ont proposé l'utilisation de méthodes de clustering comme dans la méthode précédente mais la différence réside dans le fait qu'une fois la population est classée (en utilisant un algorithme de k-means par exemple), les K solutions les plus proches du centroïde seront évaluées par la fonction de fitness originale et constituent la base de l'apprentissage du RN. Enfin, les individus des générations restantes se voient attribuer une valeur de fitness approximative en utilisant le RN. Les travaux dans [47], utilisent une approche similaire ou un ensemble de **RNs** est employé pour l'approximation de la valeur de fitness. Pour chaque génération après avoir classé les individus, un ensemble de **RN** est créé est entraîné (un **RN** pour chaque classe). La limitation de cette approche est l'apprentissage répété des **RNs**, qui consomme un temps de calcul considérable. De plus, dans certains cas, l'approximation de la valeur de fitness est erronée.

Nous pouvons citer aussi les travaux dans [51], qui utilisent que la technique des **RNs** pour estimer la valeur de fitness de la population. Il y a également les travaux de Nain et Deb [55], ou les auteurs proposent d'utiliser un contrôle de l'évolution pour améliorer les résultats du RN, en alternant entre l'évaluation directe et l'approximation de la valeur de fitness par les **RNs**. Cette méthode consiste a utiliser l'AG avec une évaluation directe des individus

pendant un nombre prédéfini de générations. La population issue de cette exécution forme les données d'apprentissage du **RN** qui sera utilisé pour approximer les valeurs de fitness des individus restants. L'inconvénient de cette méthode est de connaître le bon nombre de générations pour effectuer le contrôle de l'évolution.

Dans [46], une étude comparative entre deux méthodes d'approximation a été faite sur l'utilisation des **RNs** et des méthodes de Krigeage pour l'approximation des valeurs de fitness dans un AG. Les modèles approximatifs ont été entraînés par deux méthodes d'apprentissage. Dans la première l'apprentissage se fait à partir des données issues des exécutions précédentes de l'AG. Cependant, dans la deuxième méthode, le modèle approximatif est appris par les nouvelles données générées par l'optimisation.
Les résultats de ces travaux montrent que les méthodes d'approximations sont fortement recommandées pour des problèmes complexes. A partir de ces résultats, les auteurs constatent que l'utilisation de l'approximation de fitness peut dégrader la solution, surtout quand le nombre d'itération de l'algorithme est très grand.

Nous remarquons que la majorité des travaux se basent sur l'utilisation des **RNs** pour l'approximation de la valeur de fitness, il y a aussi l'intégration des algorithmes de clustering pour améliorer les résultats de ces derniers. Cela nous a mené aux travaux de monsieur Med. Benatia [21] qui a présenté une méthode qui s'appuie sur des notions de clustering pour réduire le temps consommé dans chaque itération de l'algorithme. en utilisant l'estimation de la valeur de fitness en se basant sur un **RN** appris par les données de la première population. Il a aussi introduit une fonction de fiabilité dans le calcul de la fonction objective de chaque individu Pour réduire l'erreur générée par les RN. Nous avons fait une étude sur son travail, nous avons également essayé d'implémenter sa méthode mais par manque de temps et de ressources nous avons pas pu avoir les même performances. Pour cela nous allons opter pour une autre approche en utilisant ses résultats comme une référence pour évaluer cette dernière qui est basée sur les arbres de décision et plus précisément "Random forest", cette méthode sera présentée en détails dans les prochaines sections.

3.4 Algorithme proposé

Comme mentionné auparavant, l'algorithme proposé a pour objectif de diminuer le nombre d'évaluations directes dans le but de réduire le temps d'exécution, en les remplaçant par

des estimations (approximatives) de la valeur de fitness des individus de la population. Les différentes phases de l'AG voir la génération des individus de la population initial, mutation , croisement et la sélection restent les mêmes que dans le chapitre précédent, la phase d'évaluation est remplacée par une phase d'approximation des valeurs de fitness par l'algorithme "Random Forest" qui est entraîné en utilisant les données de la population initiale. Il y a aussi l'introduction d'une nouvelle phase, celle de l'apprentissage du modèle approximatif. La nouvelle démarche de l'AG est illustrée dans le schéma de la figure 3.4.

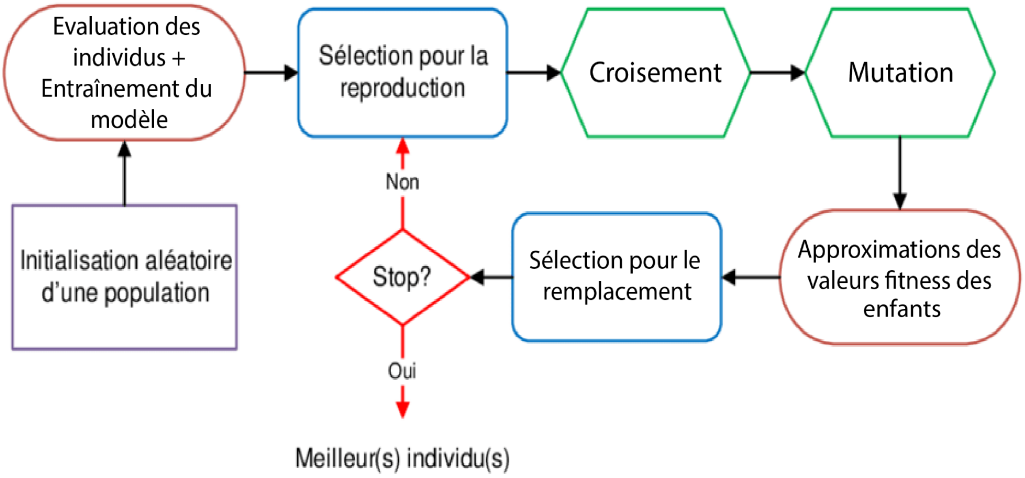

FIGURE 3.4: "Architecture de l'algorithme proposé"

3.4.1 Bibliothèque utilisée

Scikit-learn est une bibliothèque Python open source qui dispose d'outils puissants pour l'analyse et l'exploration des données. Elle est disponible sous la licence BSD et est construite sur les bibliothèques d'apprentissage automatique suivantes : NumPy, une bibliothèque permettant de manipuler des tableaux et des matrices multidimensionnels.

3.4.2 Limiter les évaluations

Nous avons déjà mentionner que le modèle approximatif doit être utilisé avec la fonction de fitness originale pour éviter les optima locaux. Pour cela nous proposons un contrôle de l'évolution par génération, donc pour chaque nombre fixé de génération appelé "control rate" la fonction originale est utilisée pour l'évaluation des individus de cette génération puis le modèle approximatif va être entraîné de nouveau en utilisant ces données. Cette opération a pour but d'assurer que le modèle approximatif ne tombe pas dans un optimum local.

3.4.3 Problème des arbres de décision

Dans une séction précédente, nous avons parler des DTs et de leur utilisation comme des modèles de régression dans les différents domaines d'optimisation. Mais ces algorithmes restent très sensibles aux données spécifiques sur lesquelles ils sont formés. Si les données d'entraînement sont modifiées, l'arbre décisionnel résultant peut être très différent et, à son tour, les prédictions peuvent être très différentes. De plus, les arbres décisionnels sont coûteux à former, présentent un risque élevé de surajustement et ont tendance à trouver des optima locaux, car ils ne peuvent pas revenir en arrière après avoir effectué une division.

3.4.4 Méthodes d'ensemble

Pour remédier à ces faiblesses, nous nous tournons vers les méthodes d'ensemble, qui illustrent la puissance de la combinaison de plusieurs modèles (ex : arbre de décision) en un seul modèle. Une méthode d'ensemble est une technique qui combine les prédictions de plusieurs algorithmes d'apprentissage automatique afin d'obtenir des prédictions plus précises que tout modèle individuel. Un modèle composé de plusieurs modèles est appelé modèle d'ensemble. Les méthodes d'ensemble les plus connues sont "le bagging", également connu sous le nom d'agrégation bootstrap, ou "le boosting".

Boosting Le boosting fait référence à un groupe d'algorithmes qui utilisent des moyennes pondérées pour transformer des modèles d'apprentissage faibles en modèles plus forts. Le boosting est basé sur le travail d'équipe. Chaque modèle exécuté détermine les caractéristiques sur lesquelles le modèle suivant se concentrera comme ilustre la Figure 3.5. Dans le boosting, comme son nom l'indique, chaque modèle apprend des autres, ce qui a pour effet de renforcer l'apprentissage.

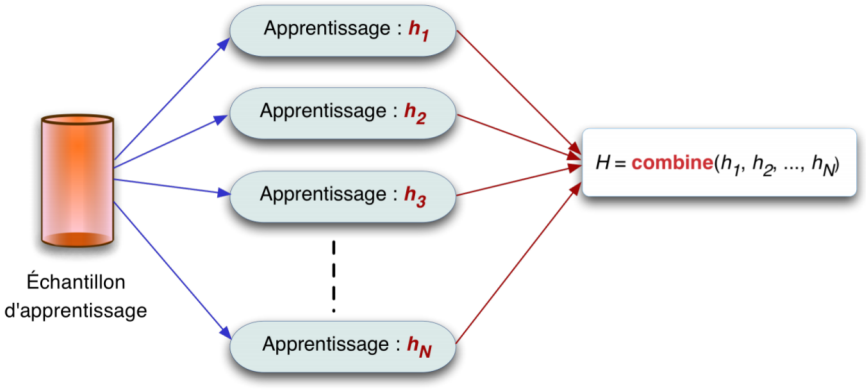

FIGURE 3.5: "Processus du Boosting de plusieurs modèles d'apprentissage"

Agrégation Bootstrap (Bagging) Bootstrap fait référence à l'échantillonnage aléatoire avec remplacement. Le bootstrap nous permet de mieux comprendre le biais et la variance de l'ensemble de données. Le Bootstrap implique un échantillonnage aléatoire d'un petit sous-ensemble de données de l'ensemble de données. Il s'agit d'une procédure générale qui peut être utilisée pour réduire la variance des algorithmes qui ont une variance élevée, généralement les DTs. Le Bagging fait tourner chaque modèle indépendamment et agrège ensuite les sorties à la fin sans préférence pour un modèle, la Figure 3.6 montre la structure de cette technique.

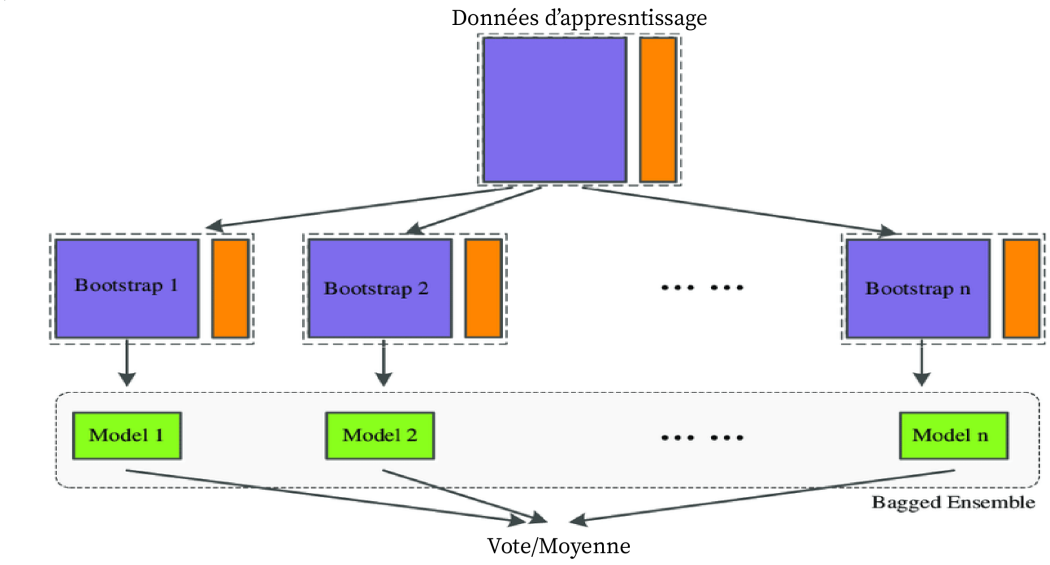

FIGURE 3.6: "Méthode d'agrégation Bootstrap"

3.4.5 Random forest

L'algorithme des "forêts aléatoires" a été proposé par Leo Breiman et Adèle Cutler [56] en 2001. Dans sa forme la plus classique, il effectue un apprentissage parallèle sur plusieurs arbres de décision construits aléatoirement et formés sur différents sous-ensembles de données. Le nombre idéal d'arbres, qui peut aller jusqu'à plusieurs centaines ou plus. Concrètement, chaque arbre de la forêt aléatoire est entraîné sur un sous-ensemble aléatoire de données selon le principe du bagging (voir la Figure 3.7), avec un sous-ensemble aléatoire de features (caractéristiques variables des données). Les prédictions sont ensuite moyennées. L'algorithme de la forêt aléatoire est connu pour être l'un des modèles "prêts à l'emploi" les plus efficaces (c'est-à-dire nécessitant peu de prétraitement des données).

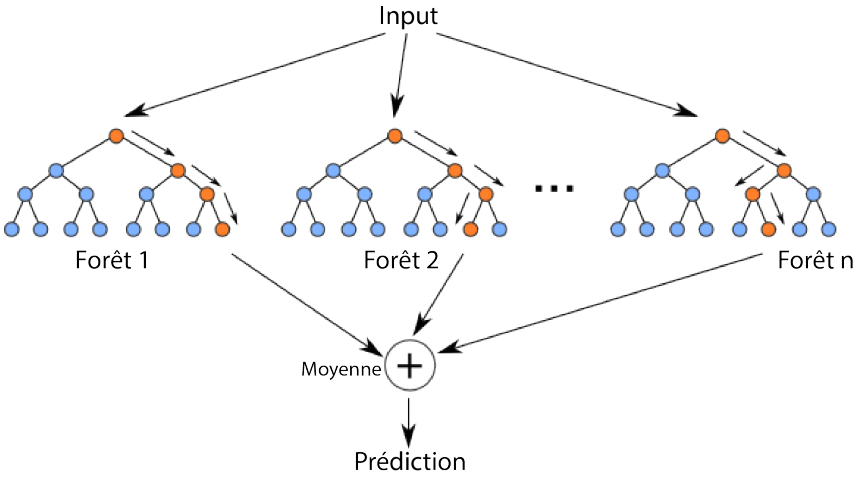

FIGURE 3.7: "Schéma de fonctionnement du modèle Random Forest"

Le RF ajoute un élément aléatoire supplémentaire au modèle, lors de la construction des arbres. Il recherche la meilleure caractéristique parmi un sous-ensemble aléatoire de caractéristiques. Il en résulte une grande diversité qui se traduit généralement par un meilleur modèle.

La principale limite de la forêt aléatoire est qu'un grand nombre d'arbres peut rendre l'algorithme trop lent et inefficace pour les prédictions en temps réel. En général, ces algorithmes sont rapides à former, mais assez lents à créer des prédictions une fois qu'ils sont formés. Une prédiction plus précise nécessite plus d'arbres, ce qui se traduit par un modèle plus lent. Dans la plupart des applications du monde réel, l'algorithme de RF est suffisamment rapide, mais il peut certainement y avoir des situations où les performances d'exécution sont importantes et où d'autres approches seraient préférables.

3.5 Implémentation de l'architecture proposée

Dans cette partie du chapitre, nous allons préparer nos données d'apprentissage en nous appuyant sur la sélection des caractéristiques. Nous allons également observer l'effet du taux de contrôle de l'évolution et tester notre modèle sur différentes tailles de bâtiments afin de fournir des solutions de qualité acceptable en un temps d'exécution considérable. Pour le nombre d'estimateurs dans l'algoritjme RF, nous avons optés pour 100-estimateurs c'est la valeur par défaut de la bibliothèque Scikit-learn. Pour cette valeur le modèle nous a donné de très bon résultats, et des simulations nous ont montré que la variation de ce paramètre n'a pas un grand impact sur la qualité du modèle.

Comme nous l'avons expliqué dans le chapitre précédent, le nombre d'individus dans la population avait un impact énorme sur le temps d'exécution, alors que son influence sur la qualité de la solution n'était pas si important. Nous avons jugé donc que le nombre d'individus devait être minimiséIl afin d'explorer efficacement l'espace de recherche. Ce compromis peut être équilibré en augmentant le nombre de générations fixées à l'algorithme. La raison en est que notre modèle d'approximation est tellement plus rapide que la fonction d'évaluation originale que même si nous augmentons le nombre de générations, nous parvenons à trouver des solutions de bonne qualité et à réduire le temps d'exécution simultanément.

3.5.1 Sélection de caractéristique

L'un des plus grands avantages de RF est sa polyvalence, il peut être utilisé pour les tâches de régression et de classification. Une autre grande qualité de l'algorithme RF est qu'il est très facile de mesurer l'importance relative de chaque caractéristique sur la prédiction. Scikit learn fournit un excellent outil pour cela, qui mesure l'importance d'une caractéristique. Il calcule ce score automatiquement pour chaque caractéristique après la formation et met les résultats à l'échelle de sorte que la somme de toutes les importances soit égale à un.

En examinant l'importance des caractéristiques, nous pouvons décider quelles sont les caractéristiques à abandonner éventuellement parce qu'elles ne contribuent pas suffisamment (ou parfois pas du tout) au processus de prédiction. C'est important car une règle générale de l'apprentissage automatique veut que plus le nombre de caractéristiques est élevé, plus le modèle risque d'être surajusté et vice versa.

La sélection de caractéristique (Feature Selection) est très essentielle car certaines caractéristiques ont généralement un impact plus important que d'autres. Cette démarche permet de supprimer des génes non significatives qui n'affectent pas trop la sortie de notre modèle mais peut en fait diminuer les performances en introduisant du bruit. Dans notre problème, une caractéristique n'est rien d'autre qu'un gène qui compose un chromosome, c'est-à-dire un individu.

En fait, moins de caractéristiques signifie généralement des modèles d'entraînement plus rapides. La sélection de caractéristiques peut également réduire le temps de calcul si les dimensions des entrées sont plus grandes. Lorsque nous conservons les caractéristiques les plus importantes, en éliminant celles que nos méthodes de sélection des caractéristiques nous conseillent de supprimer, notre modèle devient plus simple et plus facile à comprendre. La

Figure 3.8 montre les avantages de la séléction des caractéristiques, un modèle comportant 25 caractéristiques est beaucoup plus simple qu'un modèle comportant 200 caractéristiques.

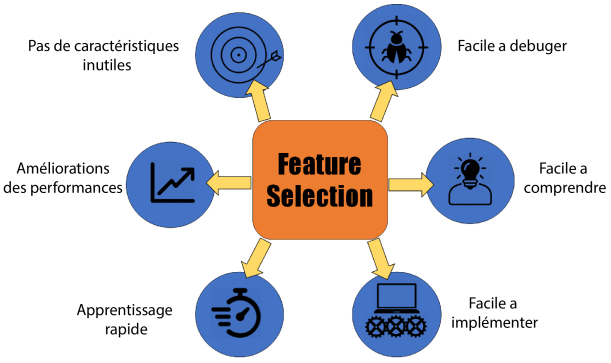

FIGURE 3.8: Schéma des avantages de la séléction des caractéristiques.

Il existe de nombreuses façons de procéder à la sélection des caractéristiques. Dans notre cas, le RF a un moyen de classer les caractéristiques par ordre d'importance et de conserver les caractéristiques les plus importantes qui fournissent une importance cumulée d'au moins 95% afin de s'assurer que notre modèle reste performant. Ce métrique d'évaluation est également connu sous le nom d'importance de Gini [57]. En utilisant un graphique comme celui-ci, nous pouvons décider où placer un seuil, en faisant un compromis entre la performance de notre modèle et le nombre de caractéristiques utilisées.

3.5.2 Taux de contrôle de l'évolution

Comme nous l'avons vu précédemment, le taux de contrôle est l'élément le plus important du contrôle de l'évolution, car il nous permet de gérer la fréquence à laquelle notre modèle est entraîné pour approximer les générations futures. En fait, afin de visualiser l'impact qu'il a sur la qualité de la solution et le temps d'exécution des calculs, nous avons étudié l'impact de ce paramètre sur le même cas d'étude que nous avons utilisé dans le chapitre précédent, celui d'un bâtiment de petites dimensions (10x10). De plus, on a fixé le nombre d'individus à 10 seulement. En contrepartie, nous avons augmenté le nombre de générations maximale à 1000 pour pouvoir mieux explorer l'espace de solutions.

Taux de contrôle	Temps d'exécution (s)	Evaluation (s)	Apprentissage (s)	Approximation (s)
2	218.94	110.03	86.73	7.76
4	127.32	61.39	43.4	11.93
6	91.64	39.51	28.88	13.12
8	86.81	36.76	21.66	13.54
10	66.26	25.14	17.65	13.86
12	64.64	24.55	14.69	14.72
14	62.51	22.82	12.49	14.72
16	59.72	19.63	10.86	14.51
18	61.31	21.25	10.66	15.69
20	53.81	16.35	9.06	15.13

TABLE 3.1: Influence du taux de contrôle sur le temps d'exécution (en seconde).

Nous avons fait varier le taux de contrôle de 2 générations à 20 avec un pas de 2 générations. Le taux de contrôle définit essentiellement la fréquence d'entraînement de notre modèle. Les individus des autres générations sont évalués à l'aide de ce modèle approximatif pour éviter l'utilisation de la fonction de fitness originale. Par exemple, dans le tableau 3.1, pour un taux de contrôle de 6, nous entraînons notre modèle avec la fonction de fitness originale toutes les 6 générations. On observe que, pour des taux de contrôle plus élevés, le temps d'exécution diminue considérablement, en raison de la diminution du temps de la phase d'évaluation.

Cependant, cette baisse du temps d'exécution s'accompagne généralement d'une baisse de la qualité due aux erreurs d'approximation produites par notre modèle. Cela peut être vérifié dans le tableau ci-dessous 3.1, car il est clair que le score de fitness diminue progressivement et que la valeur de fitness de certains objectifs n'est pas optimisée efficacement.

Taux de contrôle	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
2	17.0	99.0	100.0	26.0	39.0
4	16.0	85.0	99.0	27.0	35.25
6	17.0	93.0	100.0	20.0	39.0
8	19.0	91.0	99.0	38.0	33.25
10	20.0	100.0	100.0	45.0	33.75
12	18.0	93.0	100.0	36.0	34.75
14	18.0	80.0	99.0	41.0	30.0
16	24.0	99.0	100.0	58.0	29.25
18	21.0	96.0	100.0	45.0	32.5
20	20.0	96.0	100.0	42.0	33.5

TABLE 3.2: Influence du taux de contrôle sur la qualité des solutions.

Par conséquent, le choix du taux de contrôle dépend de notre préférence, à savoir si nous voulons trouver une solution de qualité au détriment d'un temps d'exécution court, ou vice versa.

3.5.3 Exemples de solutions

Dans cette partie, nous avons fourni quelques exemples de solutions afin de montrer comment nous avons pu surmonter le problème d'approximation de la fitness en nous appuyant sur l'hybridation. Nous nous sommes également appuyés sur l'erreur quadratique moyenne pour valider l'efficacité de notre modèle. Cette métrique a été utilisée pour comparer les valeurs de fitness pour chaque objectif, c'est à dire comparer la valeur de la fonction de fitness originale à celle du modèle approximatif.

Nous avons également mentionné que l'AG peut être utilisé avec différents critères d'arrêt. Dans nos tests, nous allons essayer de définir des critères d'arrêt en nous appuyant sur la valeur du score, afin de nous assurer que notre algorithme converge et s'arrête lorsqu'il trouve une solution que nous jugeons acceptable. Pour les deux exemples, le taux de contrôle a été fixé à 10 générations. Par contre, le nombre de générations a été fixé à 5000 afin de laisser suffisamment de temps à l'algorithme pour converger. Cependant, nous avons ajouté un critère d'arrêt basé sur le score de fitness afin de s'assurer que l'algorithme s'arrête dès qu'il trouve la solution appropriée en fonction des critères souhaités.

3.5.3.1 Bâtiment de petites dimensions

Dans un premier temps, nous avons observé l'efficacité de notre modèle sur un petit bâtiment de 100m2 (10m de longueur et 10m de largeur). Nous avons également fixé un critère de score de fitness supérieur à 40%. La figure 3.9 montre l'évolution des 4 critères de déploiement en fonction du nombre de générations. En effet, nous avons observé que l'algorithme converge après 1153 générations et qu'il s'arrête au moment où le score de fitness de sa solution valide le critère prédéfini.

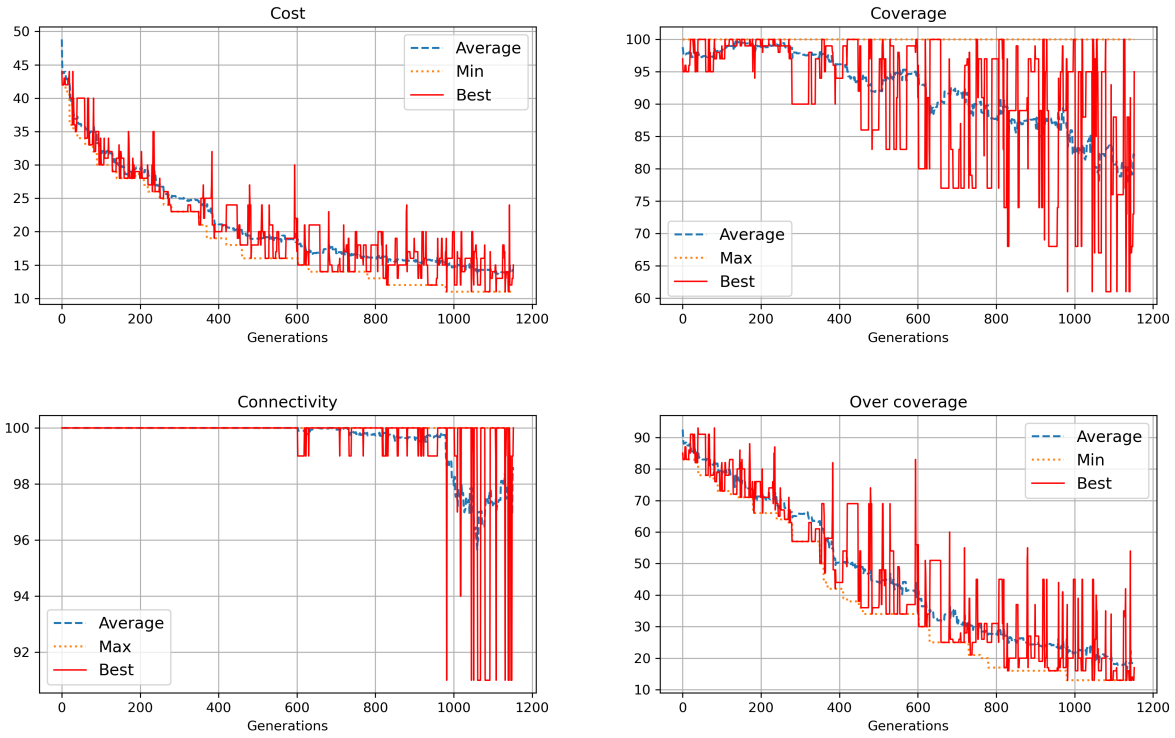

FIGURE 3.9: Évolution des valeurs de fitness pour les 4 objectifs.

Le coût et la sur-couverture sont minimisés efficacement, tandis que les valeurs de couverture et de connectivité sont très satisfaisantes puisqu'elles sont toutes deux supérieures ou égales à 95%. La figure **??** ci-dessous, fournissent une vue plus claire de l'évolution du temps d'exécution, ainsi qu'un graphique à barres qui représente les valeurs de fitness la solution pour chaque objectif (coût, couverture, connectivité et sur-couverture).

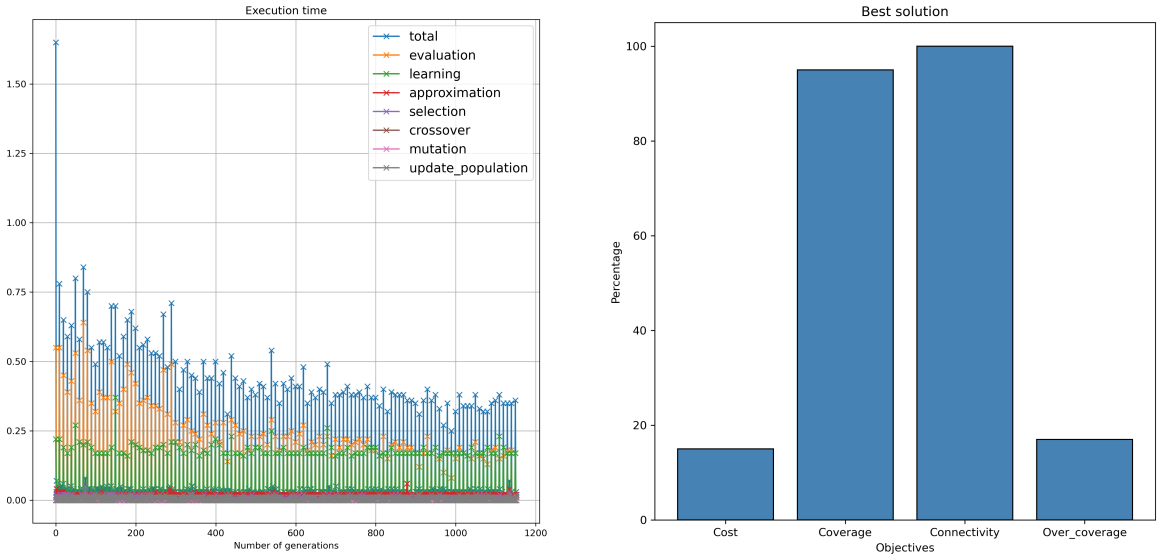

FIGURE 3.10: Graphe illustrant l'évolution du temps d'exécution et un diagramme à barres représentant les valeurs de fitness de la solution.

Après chaque phase d'apprentissage, notre modèle permet de prédire les valeurs de fitness de chaque objectif. Nous avons choisi les 10 derniers individus évalués par l'algorithme et nous avons calculé leurs valeurs de fitness en utilisant la fonction de fitness originale afin de la comparer avec notre méthode d'approximation. L'erreur calculée reste relativement faible et nous permet d'estimer les valeurs efficacement. Ces données sont présentés dans le tableau 3.3 :

Individus	Cout (%)			Couverture (%)			Connectivité (%)			Sur-couverture (%)		
	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur
1	13.0	13.78	0.78	80.0	79.73	0.27	99.0	98.82	0.18	14.0	16.93	2.93
2	13.0	13.99	0.99	82.0	82.29	0.29	100.0	98.6	1.4	16.0	18.26	2.26
3	14.0	13.69	0.31	80.0	78.11	1.89	99.0	98.72	0.28	14.0	14.01	0.01
4	14.0	14.48	0.48	88.0	84.27	3.73	100.0	98.87	1.13	16.0	18.63	2.63
5	12.0	12.83	0.83	67.0	74.27	7.27	99.0	98.36	0.64	13.0	13.85	0.85
6	11.0	12.49	1.49	61.0	71.46	10.46	91.0	96.37	5.37	13.0	13.86	0.86
7	15.0	14.0	1.0	83.0	79.74	3.26	99.0	98.83	0.17	20.0	16.07	3.93
8	17.0	16.11	0.89	98.0	92.9	5.1	100.0	99.37	0.63	28.0	27.04	0.96
9	32.0	26.52	5.48	98.0	89.64	8.36	100.0	99.19	0.81	81.0	77.8	3.20
10	15.0	14.19	0.81	88.0	82.43	5.57	100.0	98.89	1.11	16.0	15.89	0.11

TABLE 3.3: L'erreur quadratique moyenne des différents objectifs pour un bâtiment 10x10

Afin de visualiser ce résultat sur toutes les générations, nous calculons La racine de l'erreur quadratique moyenne RMSE (c'est une mesure des différences entre les valeurs prédites par un modèle ou estimateur et les valeurs réelles) pour chaque population, puis nous traçons ces valeurs en fonction du nombre de générations (voir figure 3.11. Notez que pour une phase d'apprentissage, la valeur RMSE est nulle car nous calculons les valeurs de fitness originales.

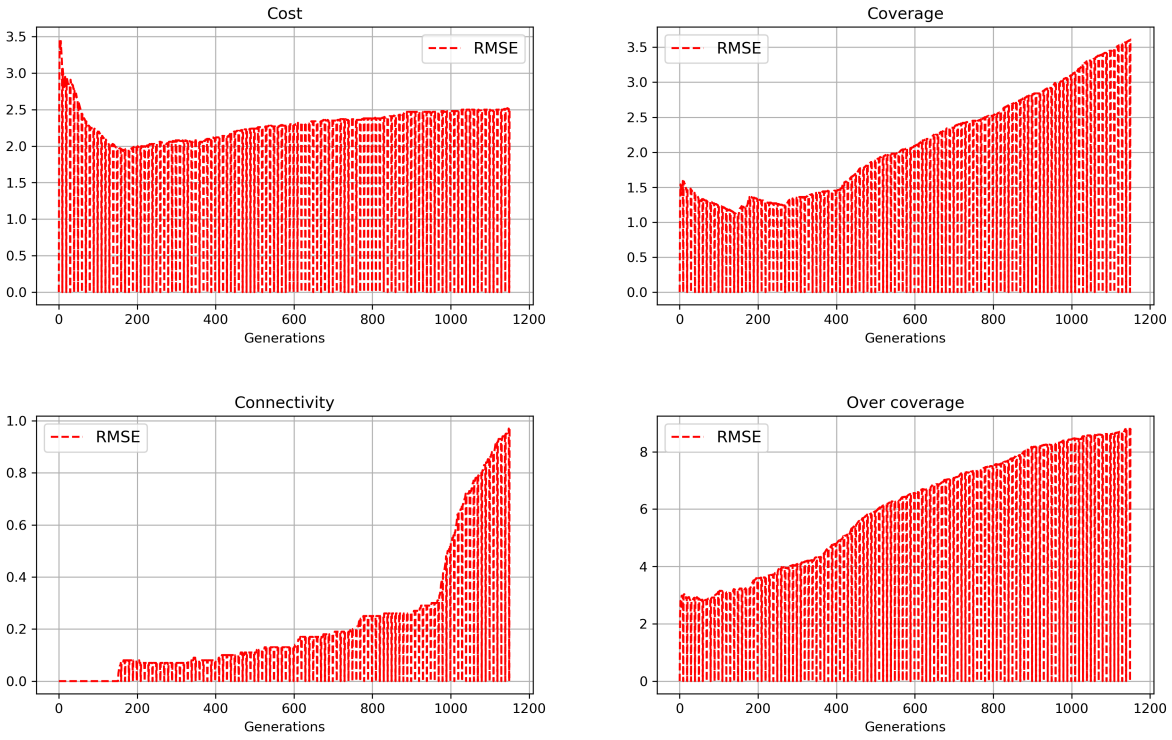

FIGURE 3.11: Évolution du RMSE en fonction des générations pour un bâtiment 10x10.

Nous avons observé que nous avons réussi à maintenir l'erreur à un niveau assez bas pour les quatre objectifs, et nous avons réussi à mettre à jour chaque individu avec des valeurs de fitness approximatives qui ont quand même réussi à converger vers une solution optimale.

Un plan de déploiement est présenté dans la figure 3.12 pour visualiser comment le placement des capteurs pour cette solution peut être mis en œuvre.

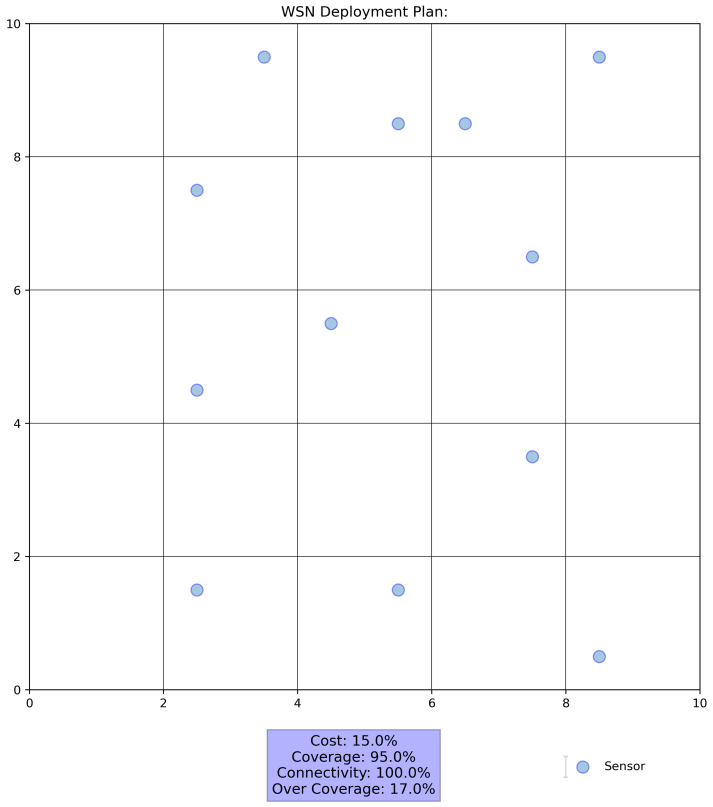

FIGURE 3.12: Plan de déploiement de la solution pour un bâtiment 10x10.

3.5.3.2 Bâtiment de grandes dimensions

Maintenant, nous procédons de la même manière qu'avant. Mais cette fois, nous essayons de trouver une solution pour un bâtiment beaucoup plus grand, par exemple un bâtiment de $20m$ de longueur et de largeur ($400m^2$). Nous avons également fixé un critère de score de fitness supérieur à 35%. Cependant, cette fois, l'algorithme a besoin de beaucoup plus de générations pour converger. En fait, 2905 générations ont été nécessaires pour que l'algorithme trouve une solution acceptable qui correspond à nos critères d'arrêt.

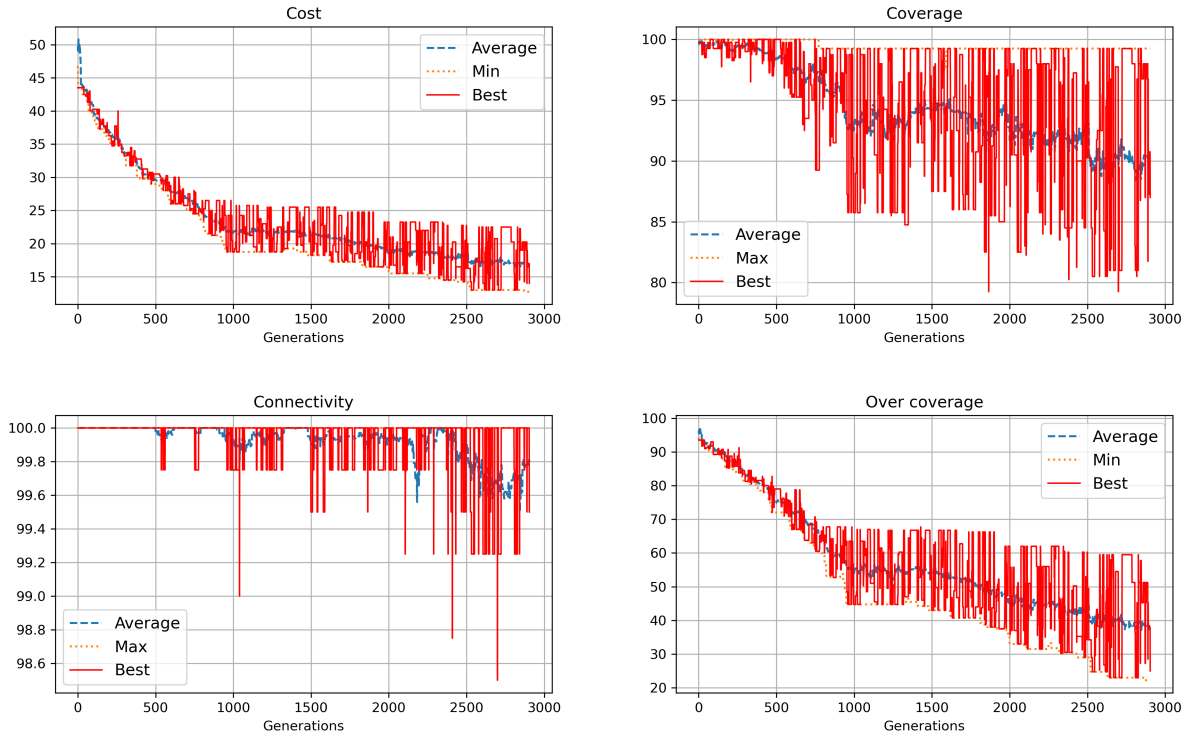

FIGURE 3.13: Évolution des valeurs de fitness pour les 4 objectifs.

De la même manière que pour la simulation précédente, nous pouvons voir sur la figure 3.13 que les 4 objectifs sont optimisés efficacement et convergent vers une solution optimale.La figure **??** ci-dessous, fournissent une vue plus claire de l'évolution du temps d'exécution, ainsi qu'un graphique à barres qui représente les valeurs de fitness la solution pour chaque objectif (coût, couverture, connectivité et sur-couverture).

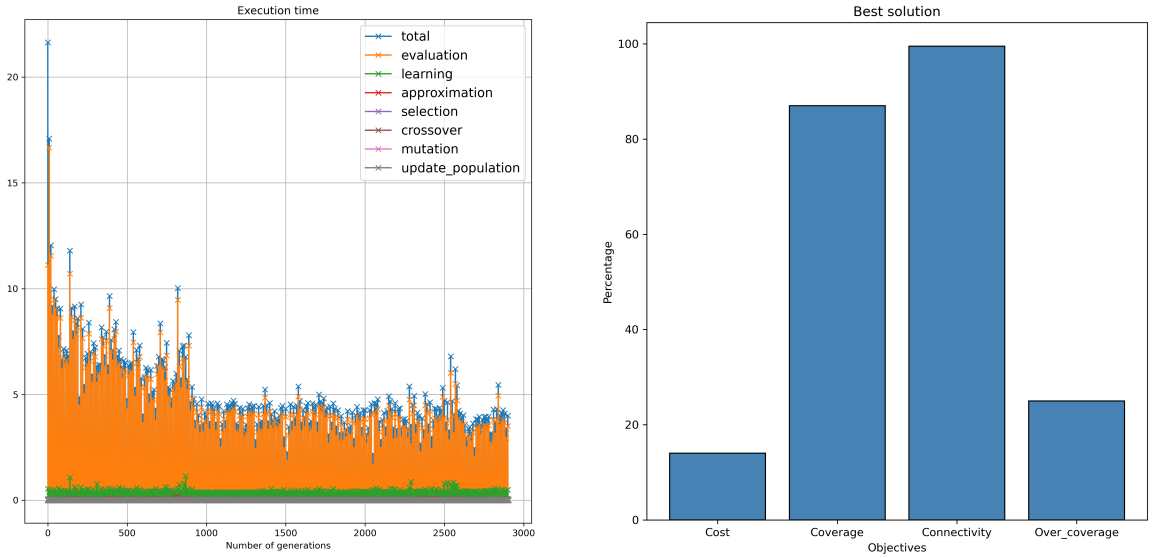

FIGURE 3.14: Graphe illustrant l'évolution du temps d'exécution et un diagramme à barres représentant les valeurs de fitness de la solution.

Comme dans le cas précédent, nous pouvons remarquer sur les résultats du tableau 3.4 que notre modèle se rapproche toujours des valeurs de fitness d'une manière acceptable. De plus, la dimensionnalité de l'entrée dépend fortement de la taille de l'individu. Bien que les dimensions du bâtiment aient augmenté, le modèle fonctionne toujours comme prévu.

	Cout (%)			Couverture (%)			Connectivité (%)			Sur-couverture (%)		
	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur	Réel	Prédiction	Erreur
1	16.75	17.68	0.93	88.0	91.66	3.66	100.0	99.97	0.03	39.0	41.7	2.7
2	16.5	16.18	0.32	84.75	87.16	2.41	99.75	99.87	0.12	38.5	35.9	2.6
3	13.0	15.18	2.18	83.25	88.84	5.59	99.5	99.74	0.24	21.5	31.48	9.98
4	18.0	16.4	1.6	94.5	88.48	6.02	100.0	99.77	0.23	42.0	35.61	6.39
5	15.75	16.12	0.37	89.25	89.95	0.7	99.25	99.51	0.26	31.75	34.79	3.04
6	16.0	15.63	0.37	87.25	86.59	0.66	100.0	99.96	0.04	36.0	34.06	1.94
7	17.0	17.68	0.68	87.25	91.66	4.41	100.0	99.97	0.03	41.5	41.7	0.2
8	16.75	15.74	1.01	88.5	88.52	0.02	100.0	99.84	0.16	38.0	33.84	4.16
9	14.75	15.18	0.43	89.75	87.3	2.45	99.75	99.79	0.04	29.5	32.13	2.63
10	14.25	14.69	0.44	81.5	85.4	3.9	99.75	99.76	0.01	29.75	30.69	0.94

TABLE 3.4: L'erreur quadratique moyenne des différents objectifs pour un bâtiment 20x20

En plus de cela, le RMSE est également assez faible comme le montre la figure 3.15. Cela nous renseigne sur la manière dont notre modèle s'adapte aux données. De plus, la

sélection des caractéristiques nous a permis de sélectionner uniquement les gènes qui ont eu un impact sur la sortie des données d'entraînement. Par exemple, au lieu d'utiliser 400 entrées (taille de l'individu), nous effectuons une sélection des caractéristiques avant chaque phase d'apprentissage pour éliminer les caractéristiques non essentielles et rendre l'approximation encore plus rapide.

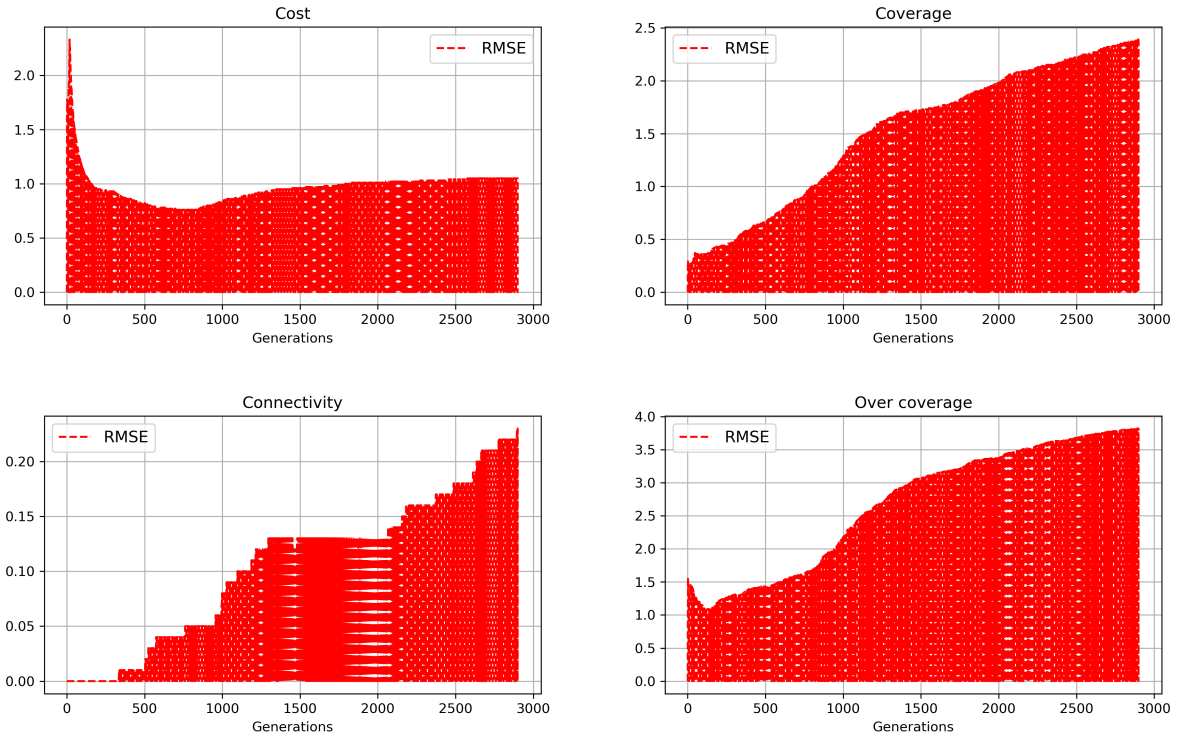

FIGURE 3.15: Évolution du RMSE en fonction des générations pour un bâtiment 20x20.

Un plan de déploiement est présenté dans la figure 3.16 pour visualiser comment le placement des capteurs peut être mis en place.

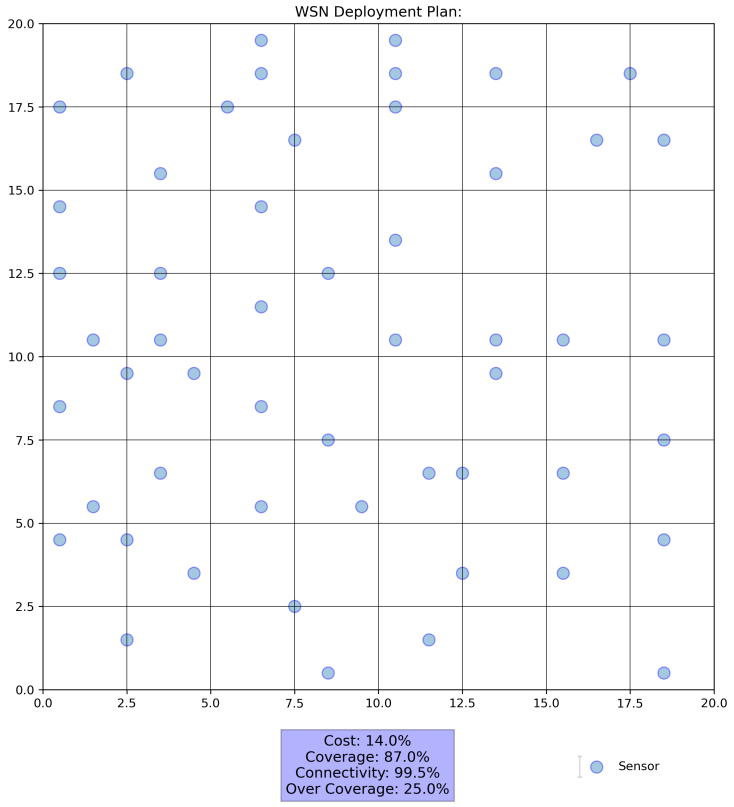

FIGURE 3.16: Plan de déploiement de la solution pour un bâtiment 20x20.

3.6 Résultats et discussion

Enfin, nous allons comparer les solutions calculées dans le chapitre précédent à celles obtenues par la méthode d'hybridation que nous avons mise en œuvre. L'architecture proposée a été utilisée pour calculer des solutions pour les mêmes tailles de bâtiments que celles testées précédemment afin de juger de l'amélioration que nous avons apportée.

Les deux tableaux 3.5 et 3.6 montrent que notre méthode hybride fournit non seulement des résultats satisfaisants en termes de qualité, mais parvient également à calculer la solution en moins de temps.

Dimensions	Coût (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)
$64m^2$	12.5	85.94	100.0	12.5	40.23
$100m^2$	15.0	95.0	100.0	17.0	40.75
$225m^2$	12.44	87.56	100.0	13.78	40.33
$400m^2$	14.0	87.0	99.5	25.0	36.88

TABLE 3.5: Valeurs de fitness calculées par la méthode d'hybridation.

Dimensions	Temps d'exécution (s)	Evaluation (s)	Apprentissage (s)	Approximation (s)
$64m^2$	33.1	6.73	11.8	9.97
$100m^2$	78.42	30.12	21.31	17.23
$225m^2$	549.2	355.94	125.9	40.9
$400m^2$	1557.78	1333.63	129.38	39.54

TABLE 3.6: Temps d'exécution de la méthode d'hybridation pour differentes dimensions.

Le tableau 3.7 permet de comparer la qualité des solutions et le temps de calcul avec les résultats du chapitre précédent, il est clair que notre méthode hybride est meilleure non seulement en termes de convergence rapide, mais aussi en termes de qualité. L'algorithme génétique est un outil puissant pour résoudre des problèmes NP-difficile, surtout lorsqu'il est combiné avec différentes techniques d'hybridation. Par conséquent, nous avons réussi à résoudre le problème d'approximation de la valeur de fitness pour ce cas, et ainsi à réduire le temps consommé par la phase d'évaluation de l'AG.

Dimensions	Méthode	Cout (%)	Couverture (%)	Connectivité (%)	Sur-couverture (%)	Score (%)	Temps d'exécution (s)
$64m^2$	AG	9.38	78.12	98.44	1.56	41.41	298.29
	AG + RF	**12.5**	**85.94**	**100.0**	**12.5**	**40.23**	**33.1**
$100m^2$	AG	17.0	98.0	100.0	26.0	38.75	419.94
	AG + RF	**15.0**	**95.0**	**100.0**	**17.0**	**40.75**	**78.42**
$225m^2$	AG	23.56	100.0	100.0	53.33	30.78	1576.78
	AG + RF	**12.44**	**87.56**	**100.0**	**13.78**	**40.33**	**549.2**
$400m^2$	AG	25.25	97.0	100.0	60.0	27.94	6038.13
	AG + RF	**14.0**	**87.0**	**99.5**	**25.0**	**36.88**	**1557.78**

TABLE 3.7: Tableau récapitulatif des résultats.

La méthode d'approximation sur laquelle nous nous sommes appuyés est un modèle d'apprentissage automatique qui nous aide à évaluer les valeurs de fitness de chaque individu de manière rapide et efficace. Nous avons pu surmonter le problème rencontré au chapitre 2, où des données d'entrée de plus grande dimension créaient un espace de recherche plus vaste pour notre algorithme génétique, au point qu'il ne parvenait pas à converger vers une solution optimale. Cependant, notre méthode hybride a été en mesure d'améliorer le processus d'évaluation de la valeur de fitness en s'appuyant sur une approche de régression où l'algorithme RF a pu utilier des données d'apprentissage pour voir estimer ces valeurs tout en minimisant l'erreur. Cette combinaison avec la fonction de fitness originale était assez essentielle pour éviter que l'algorithme ne converge vers un faux optimum.

3.7 Conclusion

Afin de remédier au problème de lenteur des méthodes évolutionnaires utilisées et causé par la phase d'évaluation qui consomme beaucoup de temps de calcul, nous avons présenté dans ce chapitre différentes méthodes d'hybridation qui peuvent améliorer les performances des méthodes classiques.

Un état de l'art de ces méthodes a été présenté dans ce chapitre. Ensuite, nous avons présenté l'architecture de la méthode d'hybridation proposée. Cette méthode est basée sur l'apprentissage automatique supervisé et les méthodes d'ensemble. L'objectif est de réduire le temps consommé à chaque itération de l'algorithme. Ceci est assuré par le modèle de forêt aléatoire, qui a été capable de résoudre le problème d'approximation de la fitness. Dans un premier temps, nous avons commencé par un processus de sélection des caractéristiques qui ne retient que les caractéristiques importantes, puis le modèle a été entraîné fréquemment en s'appuyant sur la fonction de fitness originale afin de s'assurer que notre algorithme ne reste pas bloqué dans un faux optimum. Cependant, ce modèle a été préféré à d'autres modèles car il nécessite moins de données d'entraînement et possède une forte capacité d'adaptation aux nouvelles données.

Nous avons également présenté différents résultats de simulations pour de multiples dimensions de bâtiments afin de nous assurer que le modèle fonctionne pour toutes les tailles et nous sommes arrivés à la conclusion que notre nouvelle méthode est plus pereormante non seulement en termes de temps de calcul, mais aussi en termes de qualité.

Conclusion générale et perspectives

Ce projet de fin d'étude visait à trouver une technique d'hybridation pour améliorer les performances d'une méthode d'optimisation multi-objectifs qui vise un déploiement efficace d'un RCSF tout en s'assurant que ses performances soient satisfaisantes en termes de qualité et de temps d'exécution.

Afin de résoudre ce problème, nous avons commencé par modéliser mathématiquement les différents critères RCSF. Nous avons ensuite examiné les différentes méthodes de résolution trouvées dans la littérature. Le problème du déploiement est considéré comme un problème d'optimisation multi-objectif de complexité NP-hard. Nous avons donc choisi une des méthodes les plus adaptées à ce problème qui est l'algorithme génétique. Nous avons pu adapter l'algorithme à ce problème particulier et avons introduit le concept de fonction de fitness pour comparer la qualité des solutions possibles concernant les critères de déploiement tels que le coût, la connectivité, la couverture et la sur-couverture. Nous avons pu sélectionner l'algorithme multi-objectif approprié (SPEA-II) et définir la meilleure combinaison de paramètres génétiques à utiliser pour notre étude de cas.

Ensuite, nous avons commencé par observer et étudier les temps consommés par chaque opération de l'algorithme génétique durant toutes les générations. Nous avons remarqué que la phase d'évaluation des valeurs de fitness est la plus gourmande en terme de temps de calcul. Nous avons conclu que l'algorithme génétique peut être encore amélioré en s'appuyant sur des métaheuristiques hybrides. Il est donc envisageable d'avoir une approximation efficace en minimisant le nombre d'évaluations de fitness afin de trouver une solution acceptable.

Pour pallier ce problème, nous avons proposé de faire une approximation de la valeur de fitness grâce à un modèle d'apprentissage automatique basée sur la méthode d'ensemble. Les différentes phases de l'AG restent les mêmes. Cependant, la phase d'évaluation est remplacée

par une phase d'approximation des valeurs de fitness par l'algorithme "Random Forest". Il y a aussi l'introduction d'une nouvelle phase, celle de l'apprentissage du modèle approximatif.

Dans un premier temps, nous avons commencé par un processus de sélection des caractéristiques qui ne retient que les caractéristiques importantes, puis le modèle a été entraîné fréquemment en s'appuyant sur la fonction de fitness originale afin de s'assurer que notre algorithme ne reste pas bloqué dans un faux optimum. En revanche, ce modèle a été préféré à d'autres modèles car il nécessite moins de données d'entraînement et possède une forte capacité d'adaptation aux nouvelles données. Nous avons également présenté différents résultats de simulations pour de multiples dimensions de bâtiments afin de nous assurer que le modèle fonctionne pour toutes les tailles et nous sommes arrivés à la conclusion que notre nouvelle méthode est plus performante non seulement en termes de temps de calcul, mais aussi en termes de qualité.

Enfin, une perspective intéressante serait l'utilisation de modèles beaucoup plus complexes en ce qui concerne la couverture et la connectivité, car on a opté pour des modèles beaucoup plus simples afin de faciliter l'implémentation. Le fait d'avoir une représentation plus compliquée peut engendrer une hausse considérable du temps d'exécution des algorithmes, donc l'hybridation peut être utilisée pour atténuer ou même éviter cet obstacle. De plus, les AGs se prêtent bien à la parallélisation. La valeur de fitness est calculée indépendamment pour chaque individu, ce qui signifie que tous les individus de la population peuvent être évalués simultanément. En outre, les opérations de sélection, de croisement et de mutation peuvent être effectuées simultanément sur les individus et les paires d'individus de la population. Cela fait de l'approche des algorithmes génétiques un candidat naturel pour une mise en œuvre réussie sur les architectures informatiques modernes que l'on peut déployer sur le cloud.

Bibliographie

[1] "La domotique, c'est quoi ?" 2015. [Online]. Available : https://www.maison-et-domotique.com/47895-la-domotique-cest-quoi/

[2] "What is a smart building ?" 2011. [Online]. Available : https://buildingefficiencyinitiative.org/articles/what-smart-building

[3] "Smart building : c'est quoi ?" 2018. [Online]. Available : https://iotfactory.eu/fr/smart-building-cest-quoi/

[4] "5 key benefits of smart buildings," 2019. [Online]. Available : https://www.trueoccupancy.com/blog/5-key-benefits-of-smart-buildings

[5] I. Akyildiz, S. WY, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks : A survey," vol. 38, pp. 393–422.

[6] K. Ahmed and M. Gregory, "Integrating wireless sensor networks with cloud computing," in *2011 Seventh International Conference on Mobile Ad-hoc and Sensor Networks*, pp. 364–366.

[7] S. FELLAH, "Optimisation Multi-objectif appliquée au déploiement et à la performance des réseaux de capteurs sans fil," Theses, Université d'Oran1 - Ahmed Ben Bella, 2018. [Online]. Available : https://theses.univ-oran1.dz/these.php?id=TH4820

[8] Y. CHELAL, "Réseaux de capteurs sans fils," 2016. [Online]. Available : http://y_challal.esi.dz/wp-content/uploads/2016/05/PDF-RCSF.pdf

[9] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pp. 719–724, 2005.

[10] Successful deployment of a wireless sensor network for precision agriculture in malawi - million mafuta, marco zennaro, antoine bagula, graham ault, harry gombachika, timothy chadza, 2013. [Online]. Available : https://journals.sagepub.com/doi/full/10.1155/2013/150703

[11] H. Belmonte, "Étude, implémentation et évaluation d'un réseau de capteurs sans fil exploitant le concept de colportage de l'information," Master's thesis, Jun. 2011. [Online]. Available : https://dumas.ccsd.cnrs.fr/dumas-01076637

[12] M. P. and al., ""a survey of military applications of wireless sensor networks"," in *2012 Mediterranean Conference on Embedded Computing (MECO)*, 2012, pp. 196–199.

[13] M. Farsi, M. A. Elhosseini, M. Badawy, H. Arafat Ali, and H. Zain Eldin, "Deployment techniques in wireless sensor networks, coverage and connectivity : A survey," *IEEE Access*, vol. 7, pp. 28 940–28 954, 2019.

[14] D. He, G. Mujica, J. Portilla, and T. Riesgo, "Modelling and planning reliable wireless sensor networks based on multi-objective optimization genetic algorithm with changeable length," vol. 21, no. 2, pp. 257–300. [Online]. Available : https://doi.org/10.1007/s10732-014-9261-2

[15] M. Farsi, M. A. Elhosseini, M. Badawy, H. Arafat Ali, and H. Zain Eldin, "Deployment techniques in wireless sensor networks, coverage and connectivity : A survey," vol. 7, pp. 28 940–28 954. [Online]. Available : https://ieeexplore.ieee.org/document/8653909/

[16] A. Hossain, P. K. Biswas, and S. 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*, pp. 1–5, ISSN : 2164-7011.

[17] S. Harizan and P. Kuila, "Evolutionary algorithms for coverage and connectivity problems in wireless sensor networks : A study," pp. 257–280.

[18] I. si Hadj Mohand, "Etude et optimisation du déploiement d'un réseau de capteurs sans fil dans un baatiment intelligent," Graduation project, Ecole nationale polytechnique - Alger, 2020.

[19] C. A. Balanis, *Antenna theory : analysis and design*. John wiley & sons, 2016.

[20] M. Lott and I. Forkel, "A multi-wall-and-floor model for indoor radio propagation," in *IEEE VTS 53rd Vehicular Technology Conference, Spring 2001. Proceedings (Cat. No.01CH37202)*, vol. 1, 2001, pp. 464–468 vol.1.

[21] M. A. Benatia, "Multi-objective optimization of a network infrastructure dedicated to smart buildings," Theses, INSA de Rouen, Dec. 2016. [Online]. Available : https://tel.archives-ouvertes.fr/tel-01579381

[22] A. Berro, "Optimisation multiobjectif et stratégies d' évolution en environnement dynamique." [Online]. Available : http://www.theses.fr/2001TOU10091

[23] S. Ruder, "An overview of gradient descent optimization algorithms." [Online]. Available : http://arxiv.org/abs/1609.04747

[24] "A simplex method for function minimization." [Online]. Available : https://doi.org/10.1093/comjnl/7.4.308

[25] A branch-and-cut method for 0-1 mixed convex programming | SpringerLink. [Online]. Available : https://link.springer.com/article/10.1007/s101070050103

[26] F. Glover, "Future paths for integer programming and links to artificial intelligence," vol. 13, no. 5, pp. 533–549. [Online]. Available : https://www.sciencedirect.com/science/article/pii/0305054886900481

[27] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948 vol.4.

[28] Darwin. (1807) Origin of species | work by darwin. [Online]. Available : https://www.britannica.com/topic/Origin-of-Species

[29] I. Rechenberg, "Evolution strategy : Nature's way of optimization," in *Optimization : Methods and Applications, Possibilities and Limitations*, ser. Lecture Notes in Engineering, H. W. Bergmann, Ed. Springer, pp. 106–126.

[30] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, Inc.

[31] "Genetic programming : a paradigm for genetically breeding populations of computer programs to solve problems." [Online]. Available : https://dl.acm.org/doi/10.5555/892491

[32] J. Holland. Adaptation in natural and artificial systems : An introductory analysis with applications to biology, control, and artificial intelligence | MIT press eBooks | IEEE xplore. [Online]. Available : https://ieeexplore.ieee.org/book/6267401

[33] C. A. M. DINABANDHU BHANDARI and S. K. PAL. Genetic algorithm with elitist model and its convergence, international journal of pattern recognition and artificial intelligence. [Online]. Available : https://www.worldscientific.com/doi/abs/10.1142/S0218001496000438

[34] M. A. Benatia, M. Sahnoun, D. Baudry, A. Louis, A. El-Hami, and B. Mazari, "Multi-objective WSN deployment using genetic algorithms under cost, coverage, and connectivity constraints," vol. 94, no. 4, pp. 2739–2768. [Online]. Available : https://doi.org/10.1007/s11277-017-3974-0

[35] V. Pareto, *Cours d'économie politique : professé à l'Université de Lausanne*. F. Rouge, 1896, vol. 1.

[36] J. Brownlee, *Clever Algorithms : Nature-Inspired Programming Recipes*, 1st ed. Lulu.com, 2011.

[37] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm : NSGA-II," vol. 6, no. 2, pp. 182–197, conference Name : IEEE Transactions on Evolutionary Computation.

[38] O. Banimelhem, M. Mowafi, and W. Aljoby, "Genetic algorithm based node deployment in hybrid wireless sensor networks," vol. 5, no. 4, pp. 273–279, number : 4 Publisher : Scientific Research Publishing. [Online]. Available : http://www.scirp.org/Journal/Paperabs.aspx?paperid=39564

[39] S. Razak, "A review on the implementation of multiobjective algorithms in wireless sensor network," vol. 19, pp. 772–779.

[40] H. Zaineldin, M. Badawy, M. El-Hosseini, H. Ali, and A. Abraham, "An improved dynamic deployment technique based-on genetic algorithm (IDDT-GA) for maximizing coverage in wireless sensor networks," vol. 11.

[41] S. Harizan and P. Kuila, *Evolutionary Algorithms for Coverage and Connectivity Problems in Wireless Sensor Networks : A Study*. Singapore : Springer Singapore, 2020, pp. 257–280.

[42] L. Jun and L. Ling, "Comparative research on python speed optimization strategies," in *2010 International Conference on Intelligent Computing and Integrated Systems*, pp. 57–59.

[43] Solving NP-hard number matrix games with wisdom of artificial crowds. [Online]. Available : https://ieeexplore.ieee.org/document/7272959

[44] C. Blum, J. Puchinger, G. R. Raidl, and A. Roli, "Hybrid metaheuristics in combinatorial optimization : A survey," vol. 11, no. 6, pp. 4135–4151. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S1568494611000962

[45] "Hybrid metaheuristics." [Online]. Available : https://www.springer.com/gp/book/9783642306709

[46] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," vol. 9, pp. 3–12.

[47] Y. Jin and B. Sendhoff, "Reducing fitness evaluations using clustering techniques and neural network ensembles," in *Genetic and Evolutionary Computation – GECCO 2004*, ser. Lecture Notes in Computer Science, K. Deb, Ed. Springer, pp. 688–699.

[48] B. S. Yaochu Jin, Markus Olhofer. (2000) On evolutionary optimization with approximate fitness functions | proceedings of the 2nd annual conference on genetic and evolutionary computation. [Online]. Available : https://dl.acm.org/doi/10.5555/2933718.2933864

[49] C. Gambella, B. Ghaddar, and J. Naoum-Sawaya, "Optimization problems for machine learning : A survey," vol. 290, no. 3, pp. 807–828. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S037722172030758X

[50] X. J. Richard E. Neapolitan, "Simple linear regression," in *Probabilistic Methods for Financial and Marketing Informatics*. [Online]. Available : https://www.sciencedirect.com/topics/computer-science/simple-linear-regression

[51] A. Gaspar-Cunha and A. Vieira, "A multi-objective evolutionary algorithm using neural networks to approximate fitness evaluations." vol. 6, pp. 18–36.

[52] M. Somvanshi, P. Chavan, S. Tambade, and S. V. Shinde, "A review of machine learning techniques using decision tree and support vector machine," in *2016 International*

Conference on Computing Communication Control and automation (ICCUBEA), pp. 1–7.

[53] Y. Liang, Z. Ren, Y. Yang, A. Chen, D. Guo, and B. Pang, "A mahalanobis distance-based fitness approximation method for estimation of distribution algorithms in solving expensive optimization problems," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 1608–1613, ISSN : 2577-1655.

[54] M. A. Benatia, M. Sahnoun, A. Louis, and D. Baudry, "Hybrid meta-heuristics for the optimization of the deployment of wireless sensor networks (WSN)," number : 2105 Publisher : EasyChair. [Online]. Available : https://easychair.org/publications/preprint/ZT7w

[55] P. K. S. Nain and K. Deb, "A computationally effective multi-objective search and optimization technique using coarse-to-fine grain modeling," in *In 2002 Ppsn Workshop on Evolutionary Multiobjective Optimizaa Comprehensive Survey of Fitness Approximation in Evolutionary Computation 13 Tion.*

[56] L. Breiman, "Random forests," vol. 45, no. 1, pp. 5–32. [Online]. Available : https://doi.org/10.1023/A:1010933404324

[57] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, "A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data," vol. 10, no. 1, p. 213. [Online]. Available : https://doi.org/10.1186/1471-2105-10-213