
RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
CENTRE UNIVERSITAIRE BELHADJ BOUCHAIB D'AIN TÉMOUCHENT



Institut des Sciences
Département des Mathématiques et de l'Informatique

Mémoire

Pour l'obtention du Diplôme de Master en Informatique
Option : Réseaux et Ingénierie des Données (RID)

Présenté par :

Melle. Amira BENTABET
M. Slimane Charafeddine BENGHELIMA

L'OPTIMISATION DU DÉPLOIEMENT D'UN RCSF POUR UNE APPLICATION DE SURVEILLANCE DE SITES

Encadrant :

A. BENZERBADJ

Maître de Conférences Classe "B" C.U.B.B.A.T.

Co-Encadrant :

O. BAALA-CANALDA

Maître de Conférences HDR, Institut FEMTO-ST

Soutenu le 18/06/2019

Devant le jury composé de :

| | | |
|----------------|--------------------------------|--------------|
| Président : | M. R. SAIDI (M.C.B) | C.U.B.B.A.T. |
| Examineurs : | M. M. A. MESSAOUDI (M.A.A) | C.U.B.B.A.T. |
| Encadrant : | Ali BENZERBADJ (M.C.B) | C.U.B.B.A.T. |
| Co-Encadrant : | Oumaya BAALA-CANALDA (MCF HDR) | UTBM-France |

Dédicace

Je dédie ce modeste travail :

À la femme qui a souffert sans me laisser souffrir, qui n'a épargné aucun effort pour me rendre heureuse : mon adorable mère. À l'homme qui doit mon respect, qui a toujours été là pour moi : mon cher père. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance. Que Dieu leur offre une longue et joyeuse vie.

À mes chers frères Nabil et Hassane, à ma chère petite sœur Khaoula, à ma tante, et mes cousines

À tous mes enseignants et tous mes amis. Merci pour leur amour et leurs encouragements.

Amira

Dédicace

Je dédie ce modeste travail, en signe de reconnaissance et de gratitude, à ma grand-mère et mes chers parents qui ont sacrifié leur vie pour ma réussite et m'ont éclairé le chemin par leur conseils judicieux. J'espère qu'un jour, je pourrais leur rendre un peu de ce qu'ils ont fait pour moi. Que Dieu leur prête bonheur et longue vie.

A mon frère Amine, à mes cousins Nadir et Hamza, à mes cousines, à ma tante, et à mes oncles et leurs femmes.

Ce travail est également dédié à tous mes enseignants et à tous mes amis.

Slimane Charafeddine

Remerciements

Avant tout, nous remercions Allah le tout puissant de nous avoir donné la force et la volonté d'accomplir ce travail.

C'est avec une immense reconnaissance que nous adressons nos remerciements les plus sincères à notre encadrant, M. Ali BENZERBADJ d'avoir accepté de diriger ce travail, ses aides précieuses, son encouragement et ses conseils constructifs ont été d'une grande utilité pour bien mener ce travail.

Tout notre respect et nos remerciements vont à notre co-encadrante Mme. Oumaya BAALA-CANALDA, de l'UTBM (France), qui nous a orienté; ses conseils et ses précieux commentaires nous ont permis de surmonter certaines difficultés et de progresser dans notre travail.

Nous tenons à remercier M. R. SAIDI Maître de Conférences Classe "B" au niveau du Centre Universitaire BELHADJ Bouchaïb D'Ain Témouchent, d'avoir accepté de présider le jury et d'évaluer ce projet de fin d'études.

Nous remercions M. M. A. MESSAOUDI Maître Assistant Classe "A" au niveau du Centre Universitaire BELHADJ Bouchaïb D'Ain Témouchent, d'avoir accepté d'examiner notre travail.

Nous tenons aussi à exprimer nos remerciements à M. Mejdî KADDOUR, Professeur à l'Université d'Oran 1 Ahmed BEN BELLA, pour ses précieux conseils et orientations.

Nos remerciements vont aussi à Melle. Besma Zeddâ Doctorante en Première Année Mathématiques à l'Université Abou Bekr Belkaid Tlemcen.

Nous remercions également tous nos enseignants du département des Mathématiques et Informatique du Centre Universitaire Belhadj Bouchaïb d'Ain Témouchent, pour leur soutien et disponibilité.

Finalement, nous adressons nos plus sincères remerciements à nos familles et nos proches qui nous ont soutenu et encouragé, et ce durant toute la durée de ce projet.

Résumé

Les Réseaux de Capteurs Sans Fil (RCSFs) sont devenus au fil des années un domaine de recherche très attractif. D'ailleurs, ils ont retenu l'attention de beaucoup de chercheurs qui se sont intéressés à des problématiques soulevées par ces réseaux, telles que l'énergie, le déploiement, la couverture, la connectivité, le routage, etc. Les RCSFs se caractérisent notamment par leur aspect miniaturisé, qui les rend furtifs, et leur déploiement rapide dans des zones accessibles ou inaccessibles. Les RCSFs, qui sont considérés comme une technologie émergente, font l'objet d'une grande variété d'applications dans divers domaines tels que le domaine militaire, la santé, le transport, l'agriculture, etc.

Dans ce mémoire, nous abordons le problème du déploiement déterministe des nœuds capteurs d'un RCSF dédié à la surveillance des sites stratégiques tel qu'un site pétrolier, nucléaire ou tout édifice sensible (gouvernemental ou autre). Nous avons scindé les nœuds du réseau en deux catégories. Les nœuds capteurs sentinelles (Sentinel Nodes -SN-), déployés à la frontière du site afin de détecter toute intrusion, et les nœuds relais (Relay Nodes -RN-) déployés à l'intérieur du site et qui sont chargés de relayer les alertes générées par les SNs jusqu'au Sink. Notre premier objectif est de trouver les emplacements optimaux des nœuds afin d'assurer la couverture barrière (frontière du site), une connectivité à moindre coût, en termes de nombre de sauts traversés par une alerte, depuis chaque nœud sentinelle jusqu'au sink. Cette limitation du nombre de sauts traversés va avoir un effet positif sur l'énergie consommée ainsi que sur la latence (le temps de bout en bout) des alertes. Le second objectif est de réduire le nombre de relais déployés afin de réduire le coût de déploiement.

Le problème a été formalisé sous forme de deux BILP (Binary Integer Linear Program) que nous avons résolus en utilisant le solveur *CPLEX*. Nous avons aussi évalué notre solution à l'aide du simulateur *CASTALIA*. Les résultats obtenus sont très satisfaisants, en termes de nombre de sauts traversés par une alerte, et par conséquent en termes de latence et d'énergie.

Mots clés : Réseaux de Capteurs Sans Fil, Déploiement déterministe, Couverture, Connectivité, Optimisation combinatoire Multi-objectif, Programmation linéaire.

Abstract

Wireless Sensor Networks (WSNs) have become over the years a very attractive field of research. In fact, they had the attention of many researchers who have been interested in issues raised by these networks, such as energy, deployment, coverage, connectivity, routing, etc. WSNs are particularly characterized by their miniaturized aspect, which makes them stealthy, and have rapid deployment in accessible or inaccessible zones. WSNs, which are considered an emerging technology, have a wide variety of applications in various fields such as military, health, transportation, agriculture, etc. In this thesis, we address the problem of the deterministic deployment of sensor nodes of a WSN dedicated to the monitoring of strategic sites such as an petroleum or nuclear sites or any sensitive building (government or other). We split the network nodes into two categories. Sentinel Sensor Nodes -SN-, deployed at the site boundary to detect intrusion, and Relay Nodes -RN- deployed within the site and relaying alerts generated by SNs up to Sink. Our first objective is to find the optimal node locations in order to assert the barrier coverage (site boundary), a lower cost connectivity, in terms of the number of hops crossed, from each sentinel node to the sink. This limitation on the number of hops crossed by an alert will have a positive effect on the consumed energy as well as the latency (end-to-end delay) of the alerts. The second objective is to reduce the number of deployed relays to reduce the cost of deployment. The problem was formalized as two BILP (Binary Integer Linear Program) that we solved using the CPLEX solver. We also evaluated our solution using the Castalia simulator. The results obtained are very satisfying, in terms of the number of hops crossed by an alert, and consequently in terms of latency and energy.

Keywords: Wireless Sensor Networks, Deterministic Deployment, Coverage, Connectivity, Multi-objective Combinatorial Optimization, Linear Programming.

ملخص

اصبحت شبكات وحدات الاستشعار اللاسلكية مجالا جذابا جدا للبحث في السنوات الاخيرة . في الواقع ، لقد لفتت انتباه العديد من الباحثين المهتمين بالإشكاليات التي تثيرها هذه الشبكات مثل الطاقة المستهلكة ، كيفية نشرها، التغطية ، الاتصال ، التوجيه ، الخ. تتميز هذه الشبكات بشكل خاص بحجمها الصغير مما يجعلها سهلة الاخفاء ، ويضمن نشرها السريع في مناطق يمكن الوصول اليها او حتى يتعذر الوصول اليها. شبكات وحدات الاستشعار اللاسلكية، التي تعتبر تقنية رائدة، تقدم مجموعة واسعة من التطبيقات في مختلف المجالات منها العسكرية والصحة والنقل والزراعة وغيرها.

في هذه الأطروحة ، نعالج مشكلة النشر المحدد لوحدات الاستشعار اللاسلكية المخصصة لمراقبة المواقع الاستراتيجية مثل موقع نفطي أو نووي أو أي مبنى حساس (حكومي أو غير ذلك). لقد قمنا بتقسيم وحدات الاستشعار إلى فئتين. وحدات الاستشعار المخصصة للحراسة ، والتي يتم نشرها على حدود الموقع للكشف عن أي محاولة تسلل. وتوجد أيضا وحدات الترحيل (إيصال المعلومات) الموضوعة داخل الموقع والمهتمة فقط بنقل الإنذارات و الاشعارات التي تم رصدها و ارسالها من طرف وحدات الاستشعار المخصصة للحراسة وكذا ضمان وصولها إلى جهاز التجميع .

هدفنا الأول هو العثور على المواقع المثلى لوضع وحدات الاستشعار من أجل الحصول على تغطية شاملة وكاملة لحدود الموقع ،توفير اتصال منخفض التكلفة ، من حيث عدد القفزات (التي تدل على عدد الوحدات التي يتم العبور من خلالها)، وصولا الى جهاز التجميع. تقييد و تحديد عدد القفزات المقطوعة سيكون له تأثير إيجابي على الطاقة المستهلكة وكذلك الوقت اللازم لتوصيل الإنذارات . الهدف الثاني هو تقليل عدد وحدات إيصال المعلومات لتقليل تكلفة النشر.

تمت صياغة المشكلة على شكل ٢ يبيع التي تم حلها باستخدام الحاسب ثعبيخ. قمنا أيضا بتقييم النتيجة المتحصل عليها باستخدام برنامج المحاكاة ثستلا . النتائج التي تم الحصول عليها مرضية للغاية ، من حيث عدد القفزات التي عبرها الإنذار ، وبالتالي من حيث الوقت اللازم لتوصيله والطاقة المستهلكة.

الكلمات المفتاحية : شبكات الاستشعار اللاسلكية ، النشر المحدد ، التغطية ، التوصيل ، التحسين التوافقي متعدد الأهداف ، البرمجة الخطية.

Table des matières

| | |
|---|----------|
| Introduction | 1 |
| I Optimisation du Déploiement des RCSFs | 3 |
| Partie 1 : Les Réseaux de Capteurs Sans Fil | 3 |
| I.1 Introduction | 3 |
| I.2 Nœud capteur | 3 |
| I.2.1 Définition d'un nœud capteur | 3 |
| I.2.2 Architecture d'un nœud capteur | 4 |
| I.2.2.1 Partie Matériel | 4 |
| I.2.2.2 Partie Système d'exploitation | 6 |
| I.2.3 Modèles de nœuds capteurs | 7 |
| I.3 Réseaux de Capteurs Sans Fil | 7 |
| I.3.1 Architecture d'un RCSF | 8 |
| I.3.2 Les standards de communication pour les RCSFs | 9 |
| I.3.2.1 IEEE 802.15.4 | 9 |
| I.3.2.2 Bluetooth Low Energy | 9 |
| I.3.2.3 Sigfox | 9 |
| I.3.2.4 LoRa | 10 |
| I.3.3 Domaines d'application | 10 |
| I.3.3.1 Le domaine militaire | 11 |
| I.3.3.2 La surveillance de l'environnement | 11 |
| I.3.3.3 La surveillance sanitaire | 12 |
| I.3.3.4 Les applications domestiques | 12 |
| I.3.4 Types de Réseaux de Capteurs Sans Fil | 12 |
| I.3.4.1 RCSFs terrestres | 12 |
| I.3.4.2 RCSFs souterrains | 12 |

| | | |
|---------|--|----|
| I.3.4.3 | RCSFs sous-marins | 13 |
| I.3.4.4 | RCSFs multimédia | 13 |
| I.3.4.5 | RCSFs avec nœuds mobiles | 13 |
| I.3.5 | Caractéristiques des RCSFs | 14 |
| I.3.5.1 | Forte densité des nœuds | 14 |
| I.3.5.2 | Ressources limitées | 14 |
| I.3.5.3 | L'absence d'infrastructure (auto-organisation du réseau et la topologie dynamique) | 14 |
| I.3.5.4 | Mode de communication | 14 |
| I.3.5.5 | Sécurité limitée | 14 |
| I.3.6 | Facteurs de conception d'un RCSF | 15 |
| I.3.6.1 | Tolérance aux pannes | 15 |
| I.3.6.2 | Facteur d'échelle ou scalabilité | 15 |
| I.3.6.3 | Coût de production | 15 |
| I.3.6.4 | Environnement | 15 |
| I.3.6.5 | Support de transmission | 16 |
| I.3.6.6 | Consommation d'énergie | 16 |
| I.3.7 | Impact de l'environnement de déploiement sur le Canal Sans Fil | 16 |
| I.3.7.1 | Modèle de propagation en espace libre | 16 |
| I.3.7.2 | Modèle Log-distance path loss | 17 |
| I.3.7.3 | Modèle log-normal shadowing | 17 |
| I.3.7.4 | Modèle Okumura-Hata | 18 |

Partie 2 : Stratégies de déploiement des Nœuds Capteurs dans les RCSFs 20

| | | |
|---------|--|----|
| I.4 | Introduction | 20 |
| I.5 | Les méthodes de déploiement | 20 |
| I.5.1 | Déploiement aléatoire | 21 |
| I.5.2 | Déploiement déterministe | 21 |
| I.5.3 | Déploiement hybride | 21 |
| I.6 | Stratégies de placement des nœuds | 21 |
| I.6.1 | Stratégies basée sur les grilles | 21 |
| I.6.1.1 | Grille triangulaire | 22 |
| I.6.1.2 | Grille carré | 22 |
| I.6.1.3 | grille hexagonale | 23 |
| I.6.2 | Les stratégies basées sur la géométrie algorithmique | 23 |
| I.6.2.1 | Le diagramme de Voronoï | 24 |

| | | |
|-----------|--|-----------|
| I.6.2.2 | Triangulation de Delaunay | 24 |
| I.7 | Principaux objectifs du déploiement | 25 |
| I.7.1 | Couverture | 25 |
| I.7.1.1 | La couverture de zone | 26 |
| I.7.1.2 | Couverture de cible | 26 |
| I.7.1.3 | Couverture de barrière | 26 |
| I.7.2 | Connectivité | 26 |
| I.7.3 | Latence et nombre de sauts | 27 |
| I.7.4 | Energie | 27 |
| I.8 | Etat de l'art sur le déploiement des Nœuds Capteurs dans les RCSFs | 28 |
| I.9 | Conclusion | 29 |
| II | Optimisation Combinatoire : Etat de l'art | 31 |
| II.1 | Introduction | 31 |
| II.2 | Problèmes d'optimisation combinatoire | 31 |
| II.2.1 | Définition | 31 |
| II.2.2 | Quelques problèmes classiques de l'optimisation combinatoire | 32 |
| II.2.2.1 | Problème du sac à dos (Knapsack Problem) | 32 |
| II.2.2.2 | Problème du voyageur de commerce (Travelling Salesman Problem -TSP-) | 33 |
| II.2.2.3 | Problème de coloration de graphe | 33 |
| II.3 | L'optimisation combinatoire mono-objectif | 34 |
| II.4 | L'optimisation combinatoire multi-objectif | 34 |
| II.4.1 | Stratégies de prise en compte des objectifs | 36 |
| II.4.1.1 | Les méthodes non agrégées et non pareto | 36 |
| II.4.1.2 | Les méthodes agrégées | 36 |
| II.4.1.3 | Les méthode Pareto | 37 |
| II.5 | Approches classiques de résolution des problèmes d'optimisation combinatoire | 37 |
| II.5.1 | Les méthodes exactes | 38 |
| II.5.1.1 | Branch and Bound | 38 |
| II.5.1.2 | Programmation Dynamique | 40 |
| II.5.2 | Les méthodes Approchées | 41 |
| II.5.2.1 | Heuristiques | 41 |
| II.5.2.2 | Méta-heuristiques | 42 |
| II.6 | Conclusion | 46 |

| | |
|--|---------------|
| III Implémentation et évaluation des performances | 47 |
| III.1 Introduction | 47 |
| III.2 Problème de placement optimal des nœuds | 48 |
| III.3 Programmation linéaire | 49 |
| III.4 Le Solveur <i>CPLEX</i> | 50 |
| III.5 Solution proposée | 51 |
| III.5.1 BILP sans contraintes de minimisation du nombre total de nœuds relais | 52 |
| III.5.1.1 Les variables de décision | 52 |
| III.5.1.2 Fonction objectif | 53 |
| III.5.1.3 Les contraintes | 53 |
| III.5.1.4 Description des contraintes | 53 |
| III.5.1.5 Evaluation des performances du BILP | 55 |
| III.5.2 BILP avec les contraintes de minimisation du nombre total de nœuds relais | 56 |
| III.5.2.1 Les variables de décision | 56 |
| III.5.2.2 Fonction objectif | 56 |
| III.5.2.3 Les contraintes | 56 |
| III.5.2.4 Description des contraintes | 57 |
| III.5.2.5 Evaluation des performances BILP | 57 |
| III.5.3 Temps d'exécution des BILPs | 58 |
| III.6 Validation des résultats par simulation | 58 |
| III.6.1 Evaluation du nombre de sauts moyen | 59 |
| III.6.2 Evaluation de la latence dans le réseau | 60 |
| III.6.3 Evaluation de la consommation d'énergie dans le réseau | 61 |
| III.7 Conclusion | 62 |
| Conclusion générale et perspectives | 63 |
| Annexes | 66 |
| Architecture de l'application | 66 |
| IBM ILOG CPLEX Optimization Studio | 67 |
| 1 Présentation | 67 |
| 1.1 Modes d'utilisation | 68 |
| 2 Installation sous Windows | 69 |
| 3 Installation sous UNIX, Linux ou Macintosh | 69 |

TABLE DES MATIÈRES

| | | |
|-------|--|-----------|
| 4 | Le langage OPL (Optimization Programming Language) | 70 |
| 4.1 | Exemple : création et résolution d'un modèle à 2 variables et 2 contraintes | 71 |
| 4.1.1 | Notions de base de la syntaxe OPL | 71 |
| 4.1.2 | Création du projet et du modèle | 71 |
| 4.1.3 | Résolution | 72 |
| | MATLAB (MATrix LABoratory) | 74 |
| 1 | Présentation | 74 |
| 1.1 | Interface de MATLAB | 75 |
| 2 | Les particularités de MATLAB | 75 |
| | Bibliographie | 75 |

Table des figures

| | | |
|-------|---|----|
| I.1 | Exemple d'un Réseau de Capteurs Sans Fil [44]. | 4 |
| I.2 | Composants de base d'un nœud capteur [7]. | 5 |
| I.3 | Architecture d'un Réseau de Capteurs Sans Fil. | 8 |
| I.4 | Quelques domaines d'application pour les RCSFs [57] | 11 |
| I.5 | Les méthodes de déploiement. | 20 |
| I.6 | Modèle triangulaire [4]. | 22 |
| I.7 | Modèle carré [37] | 23 |
| I.8 | Modèle d'hexagone [4]. | 23 |
| I.9 | Le diagramme de Voronoï [37]. | 24 |
| I.10 | La Triangulation de Delaunay [37]. | 25 |
| I.11 | Delaunay-Voronoi [8] | 25 |
| II.1 | Exemple de cycle Hamiltonien | 33 |
| II.2 | Exemple de coloration de graphe | 34 |
| II.3 | Approche Interactive [54]. | 36 |
| II.4 | Classification des méthodes de résolution | 37 |
| II.5 | Division D'un Problème en sous-Problèmes | 39 |
| II.6 | Algorithme génétique général | 45 |
| III.1 | Discrétisation sous forme de grille carré de l'espace de déploiement des nœuds. | 48 |
| III.2 | Modèle de surveillance | 51 |
| III.3 | Déploiement des nœuds après résolution du BILP. | 55 |
| III.4 | Déploiement des nœuds après résolution du deuxième BILP | 57 |
| III.5 | Déploiement semi-aléatoire | 59 |
| III.6 | Nombre de sauts moyen parcourus par une alerte en fonction du nombre de sources d'alerte | 60 |

TABLE DES FIGURES

| | | |
|-------|--|----|
| III.7 | Latence moyenne en fonction du nombre de sources d'alertes | 60 |
| III.8 | Énergie totale consommée dans le réseau en fonction du nombre de sources d'alerte. | 61 |
| 9 | Architecture de l'application | 66 |
| 10 | Les composants de la suite d'optimisation ILOG | 68 |
| 11 | Exemple d'un fichier modèle. | 72 |
| 12 | Résolution d'un modèle dans Cplex Studio IDE. | 73 |
| 13 | L'interface de MATLAB | 75 |

Liste des tableaux

| | | |
|-------|---|----|
| I.1 | Caractéristiques de quelques nœuds capteurs les plus courants [29]. | 7 |
| I.2 | L'exposant de l'atténuation de parcours [47]. | 17 |
| I.3 | Plages des paramètres pour le Modèle Okumura-Hata [30] | 18 |
| III.1 | Les paramètres communs aux deux BILP. | 52 |
| III.2 | Paramètres d'exécution du BILP. | 55 |
| III.3 | Statistiques obtenus après la résolution du BILP 1 | 56 |
| III.4 | Paramètres d'exécution du BILP. | 57 |
| III.5 | Statistiques obtenus après la résolution du BILP 2 | 58 |
| III.6 | Temps d'exécution des BILPs | 58 |

Introduction

De nos jours, la surveillance des zones sensibles en utilisant les RCSFs est considérée comme un domaine de recherche très prometteur. En effet, l'aspect miniature des nœuds capteurs, qui les rend dissimulables, a favorisé l'intense utilisation de ces réseaux dans le domaine de la sécurisation des sites stratégiques.

Dans ce travail, nous avons utilisé un RCSF pour surveiller un site sensible clôturé, tel qu'un site pétrolier ou nucléaire ou tout autre édifice (gouvernemental ou autre). Notre approche de surveillance consiste à déployer des nœuds capteurs tout au long de la bordure du site. Ces nœuds sont appelés des nœuds sentinelles (en Anglais, Sentinel Nodes -SNs-). Leur rôle est de générer une alerte dès qu'ils détectent une intrusion. Ces alertes doivent ensuite être relayées jusqu'au nœud puits (sink), en utilisant des nœuds relais (en Anglais, Relay Nodes -RNs-). Le nœud sink est en charge de les transmettre à un centre de prise de décision via une liaison haut débit (internet, cellulaire ou satellite).

D'une part, l'aspect miniature des nœuds fait que ces derniers soient dotés de ressources très limitées, en termes d'énergie, de puissance de calcul, de stockage et de portée de communication et de débit. D'autre part, l'aspect critique de l'application de surveillance abordée dans ce travail, exige l'acheminement des alertes en temps raisonnable, afin de ne pas mettre en danger la sécurité du site à surveiller. Par conséquent, le routage des alertes jusqu'au nœud puits doit se faire en un nombre de sauts minimal pour économiser de l'énergie et réduire la latence (le temps de bout en bout) dans le RCSF.

La solution proposée est un déploiement déterministe optimal des nœuds du RCSF. Le déploiement en question doit être fait au moindre coût (nombre de relais minimal), garantir une couverture totale des frontières du site à surveiller à travers les SNs, et assurer au moins un chemin depuis chaque nœud sentinelle et le nœud puits (sink). De surcroît, il doit assurer la transmission, au nœud puits des alertes générées par les SNs, à travers un chemin constitué d'un nombre de sauts minimal

(coût faible en énergie et en latence). A cet effet, nous avons formulé le problème de déploiement sous forme de deux BILPs (Binary Integer Linear Program) que nous avons résolus en utilisant le solveur *CPLEX*. Le résultat de la résolution des deux BILPs (Binary Integer Linear Program) en question est l'obtention des positions optimales des nœuds (SNs et RNs) au sein du RCSF. Les résultats obtenus sont satisfaisants.

Notre mémoire est organisé en trois Chapitres :

Le premier Chapitre comprend deux parties. Dans la première, nous détaillons les concepts généraux relatifs au domaine des Réseaux de Capteurs Sans Fil (RCSFs). La deuxième partie est consacrée aux stratégies de déploiement des Nœuds Capteurs dans les RCSFs. Dans le deuxième chapitre, nous présentons un état de l'art sur l'optimisation combinatoire et les approches de résolution des problèmes d'optimisation combinatoire. Le dernier chapitre est consacré à la description de la solution proposée où nous présentons notre modèle en détail ainsi que l'implémentation et l'évaluation des performances de la solution.

Optimisation du Déploiement des RCSFs

Partie 1 : Les Réseaux de Capteurs Sans Fil

I.1 Introduction

La miniaturisation croissante de l'électronique et le progrès des technologies de télécommunication permettent aujourd'hui de produire, à faible coût, de petits nœuds capteurs qui communiquent entre eux via des liaisons sans fil. Ces nœuds capteurs sont déployés aléatoirement ou d'une façon déterministe dans une zone d'intérêt pour collecter des informations du monde physique, telles que la température, la pression, la vibration, etc, et les envoyer à un type particulier de nœud appelé puits (Sink). Ces petites entités électroniques constituent les briques de base des RCSFs (Réseaux de Capteurs Sans Fil) (voir Figure I.1). Ces derniers sont utilisés par plusieurs applications militaires et civiles telles que la surveillance des champs de bataille, surveillance de l'environnement, détection biologique, diagnostics industriels, surveillance des sites startégiques (par exemple site pétrolier, site nucléaire) etc.

I.2 Nœud capteur

I.2.1 Définition d'un nœud capteur

Un nœud capteur est un dispositif de taille réduite autonome avec des ressources très limitées capable de collecter, traiter et communiquer les informations sur son environnement.

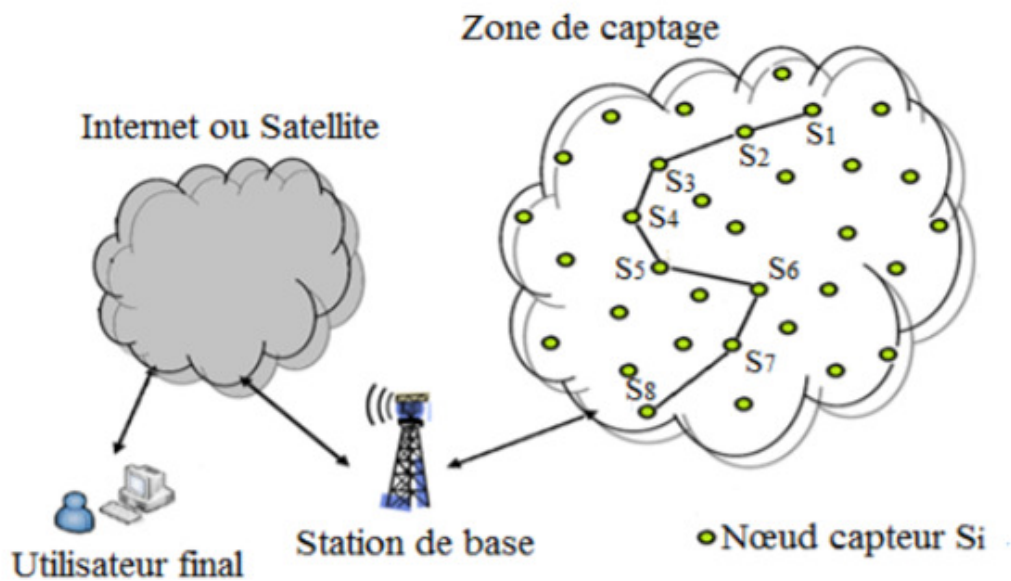


FIGURE I.1 – Exemple d’un Réseau de Capteurs Sans Fil [44].

I.2.2 Architecture d’un nœud capteur

L’architecture d’un nœud capteur comprend deux parties : matériel et système d’exploitation embarqué :

I.2.2.1 Partie Matériel

Un nœud capteur est équipé principalement de quatre unités de base, une unité de capture, une unité de traitement, une unité de communication et une unité d’alimentation en énergie [1], comme illustré dans la Figure I.2.

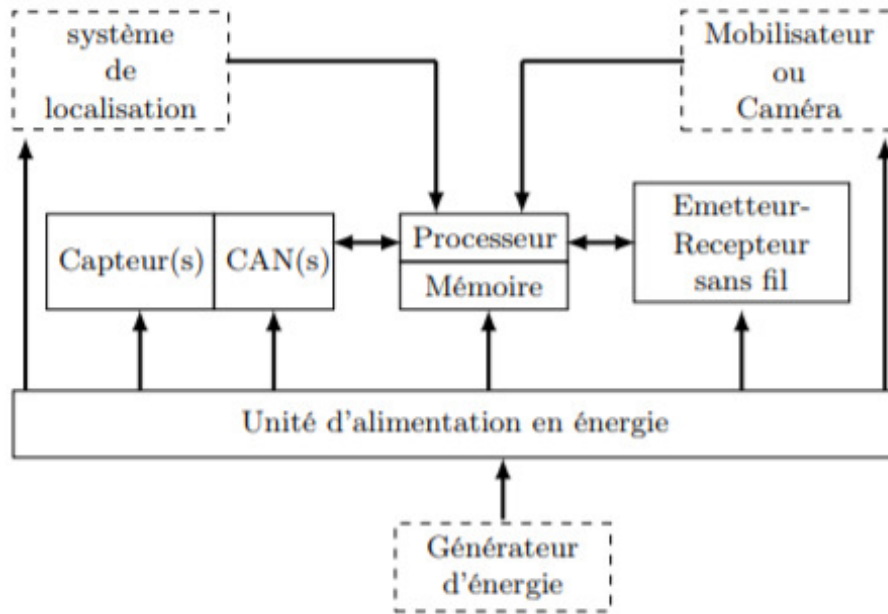


FIGURE I.2 – Composants de base d'un nœud capteur [7].

- **L'unité de capture** : est généralement composée de deux sous-unités : un capteur et un CANs (convertisseur Analogique/Numérique, en Anglais, Analog-to-Digital Converters (ADCs)). Les signaux analogiques produits par les capteurs basés sur le phénomène observé sont convertis en signaux numériques par l'ADC, puis transmis à l'unité de traitement pour pouvoir les analyser.
- **L'unité de traitement** : Elle permet le traitement des données et la gestion de la fonctionnalité d'un nœud capteur.
- **L'unité de communication** : est responsable d'effectuer toutes les émissions et réceptions de données sur le support sans fil.
- **L'unité d'alimentation en énergie** : un des composants les plus importants d'un nœud capteur est la batterie qui est, généralement, ni rechargeable ni remplaçable. Par ailleurs, les unités d'énergie rechargeables supportées par des photopiles (cellules solaires) permettent de récupérer l'énergie de l'environnement.

Un nœud capteur peut avoir également d'autres sous-unités qui dépendent de l'application telle qu'un système de localisation qui est utile pour des techniques de routage et des tâches de détection du réseau de capteurs. Un mobilisateur peut

parfois être nécessaire pour déplacer les nœuds capteurs lorsqu'il est nécessaire afin d'effectuer des tâches bien déterminées.

I.2.2.2 Partie Système d'exploitation

La configuration système requise pour les Réseaux de Capteurs comprend les fonctionnalités de base du système d'exploitation, la gestion de l'alimentation, une abstraction matérielle de détection et une pile de communication configurable, etc. Les capacités matérielles limitées imposent à ces systèmes de fonctionner avec des ressources limitées et rendent impossible l'utilisation et l'adaptation des systèmes d'exploitation traditionnels.

Plusieurs systèmes d'exploitation (OS) ont été mise en œuvre et implémentés dans les nœuds capteurs sans fil. Chacun de ces systèmes offre ses propres caractéristiques et fonctionnalités. Nous citons certaines de ces systèmes d'exploitation parmi les plus implémentés aujourd'hui dans les nœuds capteurs sans fil [22].

- **TinyOS** : est un système d'exploitation open source, flexible, conçu pour les Réseaux de Capteurs. Il a été développé à l'Université de Californie Berkely. Ce système respecte une architecture basée sur une association de composants. Sa bibliothèque de composants TinyOS comprend des protocoles réseau, des services distribués, des pilotes de capteurs et des outils d'acquisition de données, qui peuvent être adaptés aux exigences spécifiques de chaque application.
- **Contiki** : est un système d'exploitation open source léger écrit en C pour les RCSFs et développé par des chercheurs Suédois. Contiki est un système d'exploitation hautement portable construit autour d'un noyau piloté par les événements. Une installation complète de Contiki comprend des fonctionnalités telles que : noyau multitâche, multithreading préemptif, réseau TCP / IP, ainsi qu'une implémentation d'IPv6 avec le support de 6LoWPAN (IPv6 Low Power wireless Area Networks), une interface utilisateur graphique, un navigateur Web, un serveur Web personnel, un simple client Telnet, un économiseur d'écran et un réseau virtuel.
- **MANTIS** : Multimodal system for NeTworks of In-situ wireless Sensors fournit un nouveau système d'exploitation multithread pour les RCSFs. MANTIS est un système d'exploitation léger et économe en énergie. Le noyau du système est composé d'un planificateur et de pilotes de périphérique. Une pile de communication et un serveur de commandes sont fournis en tant que services de niveau utilisateur. La fonctionnalité clé du système d'exploitation MANTIS est qu'il est portable sur plusieurs plates-formes. MOS (MANTIS Operating System) prend également en charge la gestion à distance des nœuds capteurs via une program-

mation dynamique. MOS est écrit en C et prend en charge le développement d'applications en C.

I.2.3 Modèles de nœuds capteurs

les nœuds capteurs sont des dispositifs limités par une faible puissance de calcul, un espace de stockage mémoire réduit, une puissance radio et un débit faibles, ainsi qu'une source d'énergie limitée. Il existe différents modèles de nœuds capteurs sans fil qui dépendent de l'application à laquelle ils sont destinés. Le Tableau I.1 détaille les principales caractéristiques des nœuds capteurs actuels.

| Motes | MicaZ | Telos | Imote2 | WSN430 | Wavenis (Wavefront) |
|-------------------------|--------------------|--------------------|---------------------|-----------------|---------------------|
| Processeur | Atmel ATmega 128L | TI MSP430 | Intel PXA271 XScale | TI MSP430 | TI MSP430 |
| Vitesse du processeur | 16 MHz | 8 MHz | 13 - 416 MHz | 8 Mhz | 8 MHz |
| RAM | 4 Ko | 2 Ko / 10 Ko | 256 Ko | 10 Ko | 2 Ko |
| Espace programme | 128 Ko | 60 Ko / 48 Ko | 32 Mo | 48 Ko | 128 octets |
| Flash | 512 Ko | 256 Ko | 32 Mo | 1 Mo | Néant |
| Batterie | 2xAA | 2/3A | 3x AAA | PoLiFlex | 2xAA |
| Voltage | 2.7 V | 1.8 - 3.6 V | 3.2 - 4.5 V | 2.2 V | 2.4 - 6 V |
| Radio | TI CC2420 802.15.4 | TI CC2420 802.15.4 | TI CC2420 | TI CC1100 | ASIC RF Wavenis |
| Fréquence (MHz) | 2400-2483 | 2400-2483 | 2400-2483 | 315/433/868/915 | 433/868/915 |
| Débit de données (Kb/s) | 250 | 250 | 500 | 250 | 4,8 - 153 |
| Dimension (mm) | 58x32x7 | 13x26x5 | 36x48x9 | 65x40x8 | 26x20x4.5 |

Tableau I.1 – Caractéristiques de quelques nœuds capteurs les plus courants [29].

I.3 Réseaux de Capteurs Sans Fil

Un RCSF peut être défini comme un réseau de périphériques, appelés nœuds capteurs, capable de détecter l'environnement et de communiquer les informations collectées à partir du champ surveillé via des liaisons sans fil de type Radio-Fréquences (RF), acoustique, optique ou InfraRouge (IR). Les données sont transmises, éventuellement via plusieurs sauts, à un collecteur (parfois appelé contrôleur ou puits)

qui peut les utiliser localement ou qui est connecté à d'autres réseaux (par exemple, Internet) via une passerelle. Les nœuds peuvent être fixes ou mobiles. Ils peuvent être au courant de leur emplacement ou non. Ils peuvent être homogènes ou non. Ces nœuds capteurs sont caractérisés par leurs ressources limitées en termes d'énergie, de capacité de stockage, de bande passante et de portée de capture et de communication [11].

I.3.1 Architecture d'un RCSF

La Figure I.3 représente une architecture typique d'un Réseau de Capteurs Sans Fil. Ce réseau communique à travers une passerelle avec un réseau d'un autre type qui mène vers une unité de contrôle-commande.

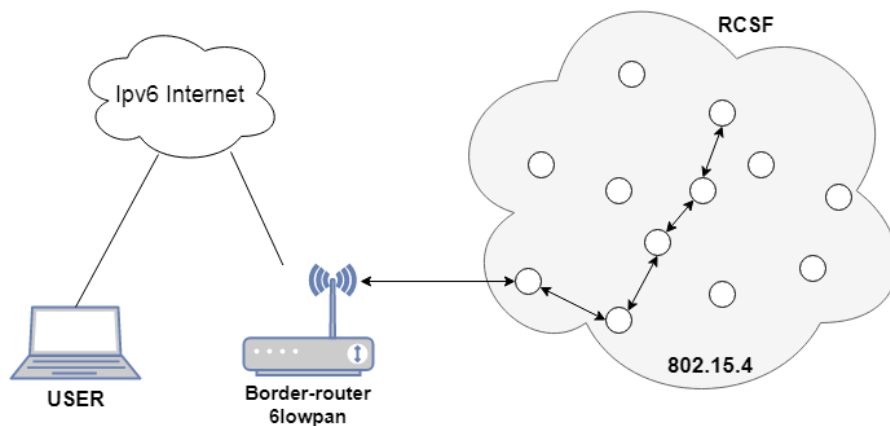


FIGURE I.3 – Architecture d'un Réseau de Capteurs Sans Fil.

- **Le champ de capture** : Les nœuds capteurs sont souvent déployés dans une zone géographique, appelée champs de capture qui définit la zone d'intérêt pour le phénomène à surveiller.
- **Les nœuds capteurs** : ils sont responsables de collecter les informations de l'environnement et acheminer les données captées au sink. Les nœuds capteurs peuvent être statiques ou mobiles.
- **6LBR** : 6LoWPAN Border Router est une passerelle qui connecte les périphériques 6LoWPAN à Internet et est également responsable de la gestion du trafic

entre les interfaces IPv6 et IEEE 802.15.4. Il possède beaucoup plus de capacités que les nœuds capteurs en termes de mémoire, de vitesse de traitement et d'énergie.

I.3.2 Les standards de communication pour les RCSFs

Les standards de communication pour les RCSFs ont été spécialement conçus pour tenir compte des ressources limitées des nœuds capteurs. Dans ce qui suit, nous décrivons brièvement les standards les plus aptes à être utilisés dans les Réseaux de Capteurs Sans Fil, à savoir IEEE 802.15.4, Bluetooth Low Energy (BLE), LORA et SIGFOX.

I.3.2.1 IEEE 802.15.4

Le standard IEEE 802.15.4 LR-WPAN (Low Rate Wireless Personal Area Network) est la norme de communication la plus utilisée dans les Réseaux de Capteurs Sans Fil. Cette technologie est caractérisée principalement par un faible coût, une faible consommation d'énergie, un faible débit de données et une courte portée de communication. La norme permet d'économiser l'énergie en mettant en œuvre le mécanisme du duty cycling (cycle d'activité de la radio), où tous les nœuds possèdent une période d'activité et une période de sommeil. Il existe deux options pour la définition des couches supérieures : les standards ZigBee et 6LoWPAN [11].

I.3.2.2 Bluetooth Low Energy

BLE (Bluetooth Low Energy) est une technologie sans fil émergente développée par le groupe SIG (Special Interest Group) pour les applications de contrôle et de surveillance à courte portée. Contrairement aux autres technologies telles que ZigBee et 6LoWPAN qui nécessitent une mise en réseau multi-sauts, BLE constitue une solution à un seul saut qui convient à une variété d'applications dans les domaines de la santé, de sport et de la sécurité [3].

I.3.2.3 Sigfox

La technologie Sigfox a été développée en 2010 par la start-up SIGFOX (Toulouse, France) qui est à la fois une entreprise et un opérateur de réseau LPWAN (Low Power Wide Area Network). Sigfox utilise les bandes de fréquences à usage libre (Industrial Scientific and Medical radio bands -ISM-), par exemple 868 MHz en Europe, 915 MHz en Amérique du Nord et 433 MHz en Asie. Cette technologie pré-

sente des niveaux de bruit très faibles, entraînant une faible consommation d'énergie et une conception à faible coût. Le débit maximal est de 100 bits/s.

Initialement, Sigfox ne prenait en charge que la communication montante (communication des nœuds capteurs aux stations de base "Sink"). Ensuite elle a évolué vers la technologie bidirectionnelle avec une asymétrie de liaison significative. Le nombre de messages sur la liaison montante est limité à 140 messages de 12 octets de charge utile par jour. Cependant, le nombre de messages sur la liaison descendante (communication des stations de base "Sink" vers les nœuds capteurs) est limité à 4 messages de 8 octets par jour, ce qui signifie que l'accusé de réception de chaque message de liaison montante n'est pas pris en charge [43].

I.3.2.4 LoRa

LoRa est une technologie de couche physique qui a été développée pour la première fois par la start-up Cycleo en 2009 (Grenoble, France) et a été achetée trois ans plus tard par Semtech (États-Unis) en 2015. Le protocole de communication basé sur LoRa appelé LoRaWAN a été normalisé par LoRa-Alliance. Cette technologie est déployée dans plusieurs pays et elle est toujours en cours de déploiement dans d'autres pays en raison des investissements de différents opérateurs mobiles (Bouygues et Orange en France, par exemple, KPN aux Pays-Bas et Fastnet en Afrique du Sud). Comme Sigfox, LoRa utilise des bandes ISM sans licence. Elle fournit la communication bidirectionnelle, et est basée sur la technologie UNB (Ultra Narrow Band) qui offre de faibles niveaux de bruit, permettant une résilience élevée aux interférences. Le débit de données LoRa est compris entre 300 bits/s et 50 kbits/s et les stations de base LoRa peuvent recevoir simultanément des messages transmis. Chaque message est de 243 octets de charge utile [43].

I.3.3 Domaines d'application

Les Réseaux de Capteurs Sans Fil se sont récemment imposés dans la communauté scientifique. Ceci est la conséquence de l'ingénierie des dispositifs de taille de plus en plus petite, qui permettent de nombreuses applications. Les RCSFs sont utilisés dans diverses domaines d'applications (voir Figure I.4), à savoir la surveillance de l'environnement, la surveillance de sites stratégiques, la santé, les applications militaires et les applications domestiques. etc [2].

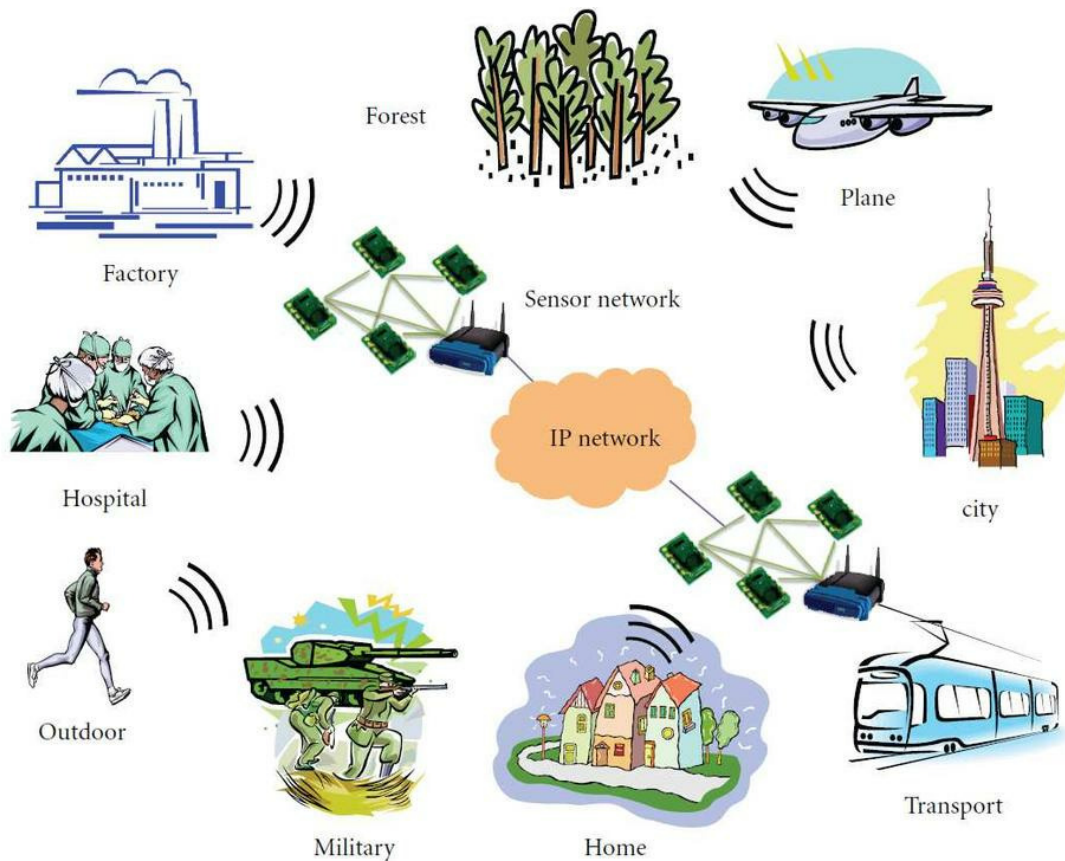


FIGURE I.4 – Quelques domaines d’application pour les RCSFs [57] .

I.3.3.1 Le domaine militaire

Les Réseaux de Capteurs ont été initialement développés pour des applications militaires. Les militaires utilisent les RCSFs pour :

- La surveillance des champs de bataille.
- La Reconnaissance des forces opposées et du terrain.
- La détection des attaques nucléaires, biologiques et chimiques.

I.3.3.2 La surveillance de l’environnement

Dans le domaine environnemental, les RCSFs peuvent être utilisés pour :

- La supervision de la qualité de l’air, la pollution.
- Suivi du déplacement des oiseaux, de petits animaux, des insectes.
- Détection d’incidents chimiques ou biologiques.

I.3.3.3 La surveillance sanitaire

Les nœuds capteurs sans fil sont utilisés dans le domaine de la médecine et d'aide aux patients. La surveillance des fonctions vitales de l'être humain serait possible grâce à des micro-capteurs qui pourront être avalés ou implantés sous la peau. Actuellement, des micro-caméras qui peuvent être avalées existent. Elles apportent une surveillance permanente des patients et une possibilité de collecter des informations physiologiques de meilleure qualité, facilitant ainsi le diagnostic de quelques maladies.

I.3.3.4 Les applications domestiques

Dans ce genre d'applications, les nœuds capteurs sont utilisés pour le "monitoring" de l'habitat. Ils peuvent être embarqués dans des systèmes domotiques de chauffage, de climatisation, d'éclairage ou de distribution d'eau et permettent de mieux gérer et d'économiser les ressources d'un immeuble et forment ainsi un environnement intelligent pour permettre à l'utilisateur final de gérer plus facilement les dispositifs domestiques localement ou à distance.

I.3.4 Types de Réseaux de Capteurs Sans Fil

Il existe plusieurs types de Réseaux de Capteurs Sans Fil qui se diffèrent généralement en fonction du domaine d'application, et selon leur environnement de déploiement ils font face à différentes contraintes et défis [48].

I.3.4.1 RCSFs terrestres

Se composent généralement d'un grand nombre (de centaines à des milliers) de nœuds à faible coût déployés dans une zone donnée. Dans ce type de réseaux, les nœuds capteurs doivent transmettre efficacement les données collectés à la station de base (Sink). Comme la charge de la batterie est limitée et généralement non rechargeable, les nœuds capteurs terrestres peuvent être équipés d'une source d'alimentation secondaire telle que des cellules solaires. Les applications les plus courantes des Réseaux de Capteurs Sans Fil terrestres sont la détection et la surveillance environnementales, la surveillance industrielle et les explorations de surface.

I.3.4.2 RCSFs souterrains

Se composent d'un certain nombre de nœuds capteurs déployés dans des grottes ou des mines ou sous terre pour surveiller les conditions souterraines. Pour trans-

mettre les informations des nœuds capteurs souterrains à la station de base (Sink), des nœuds puits supplémentaires sont situés au-dessus du sol. Ce type de réseaux font face à différents défis tels que la communication sans fil et le rechargement ou le remplacement de la batterie des nœuds déployés sous terre. La surveillance de l'agriculture, la surveillance souterraine des sols, de l'eau ou des minéraux et la surveillance des frontières militaires sont les domaines d'application des RCSFs souterrains.

I.3.4.3 RCSFs sous-marins

Un ensemble de nœuds capteurs déployés sous l'eau, peu de nœuds capteurs sont déployés dans la zone d'intérêt car ces nœuds aquatiques sont très coûteux, et des véhicules sous-marins autonomes sont utilisés pour explorer ou collecter des données à partir de ces nœuds capteurs. La communication sans fil sous-marine utilise des ondes acoustiques qui présentent divers problèmes, tels que des problèmes de bande passante, de retard de propagation et de latence élevée. Les nœuds sont équipés d'une batterie limitée qui ne peut être ni remplacée ni rechargée, ce qui nécessite des techniques de communication et de mise en réseau sous-marines éco-énergétiques. Parmi les applications des RCSFs sous-marins, figurent la surveillance de la pollution, la surveillance et l'exploration sous-marine, la surveillance sismique, la surveillance des équipements, la robotique sous-marine, etc.

I.3.4.4 RCSFs multimédia

Se composent d'un certain nombre de nœuds capteurs à faible coût équipés de caméras et de microphones. Les nœuds capteurs multimédias sont capables de stocker, de traiter et de récupérer des données multimédia telles que de la vidéo, de l'audio et des images. Ils nécessitent une bande passante élevée et ils sont caractérisés par une consommation d'énergie élevée, ils doivent fournir la Qualité de Service (QoS), le traitement de données et les techniques de compression. Ces réseaux sont déployés de manière planifiée pour permettre la surveillance et le suivi des événements et garantir la couverture. Ils peuvent être utilisés dans des applications telles que la surveillance des frontières et la surveillance du trafic routier.

I.3.4.5 RCSFs avec nœuds mobiles

Ils sont constitués de nœuds capteurs mobiles capables de se déplacer et d'interagir avec l'environnement. Les nœuds mobiles peuvent se repositionner et s'organiser au sein d'un réseau. Les RCSFs avec nœuds mobiles font face à divers défis tels que

le déploiement, la gestion de la mobilité, la localisation avec mobilité et le contrôle des nœuds mobiles, le maintien de la couverture, la réduction de la consommation d'énergie et le maintien de la connectivité réseau. Les principaux exemples d'applications des RCSFs avec nœuds mobiles sont la surveillance (environnement, habitat, sous-marin), la surveillance militaire, le suivi des cibles.

I.3.5 Caractéristiques des RCSFs

I.3.5.1 Forte densité des nœuds

les RCSFs peuvent comporter des milliers à des dizaines de milliers de nœuds. La haute densité des nœuds génère une quantité importante de données, augmente les collisions et la consommation énergétique. D'autre part, elle permet la redondance qui peut être exploitée pour renforcer la tolérance aux pannes dans ce type de réseaux.

I.3.5.2 Ressources limitées

Les nœuds capteurs sont des objets limités en termes de capacité mémoire, puissance de traitement, bande passante et énergie.

I.3.5.3 L'absence d'infrastructure (auto-organisation du réseau et la topologie dynamique)

En raison du nombre important de nœuds capteurs et leurs emplacements dans des endroits hostiles où l'intervention humaine n'est pas envisageable, l'auto-organisation peut être nécessaire. D'ailleurs, le manque d'énergie ou la destruction physique peuvent causer la mise hors service de certains nœuds. De nouveaux nœuds doivent pouvoir rejoindre le réseau initial. Le réseau doit donc être capable de s'auto-organiser et de se modifier pour qu'il puisse continuer à fonctionner.

I.3.5.4 Mode de communication

Les nœuds capteurs utilisent principalement un paradigme de communication plusieurs-à-un (many-to-one).

I.3.5.5 Sécurité limitée

Les RCSFs sont plus touchés par le problème de sécurité que les réseaux filaires classiques car, souvent, les nœuds capteurs ne sont pas protégés contre les mauvaises manipulation ou attaques. Le brouillage et les attaques de retransmission

peuvent empêcher l'opération de transmission des paquets . Cela se justifie par les contraintes et limitation physique qui font que le contrôle des données transférées est minimisé [53].

I.3.6 Facteurs de conception d'un RCSF

Plusieurs facteurs et contraintes interviennent dans la conception d'un RCSF [1].

I.3.6.1 Tolérance aux pannes

La tolérance aux pannes est la capacité de maintenir les fonctionnalités du réseau de capteurs sans interruption en dépit de l'apparition de pannes des nœuds capteurs provoquées par une défaillance ou un blocage des nœuds qui peut être engendré par plusieurs causes, notamment l'épuisement d'énergie, l'endommagement physique ou les interférences liées à l'environnement.

I.3.6.2 Facteur d'échelle ou scalabilité

Cette caractéristique traduit la capacité de maintenir la performance indépendamment de la taille du réseau car le nombre de nœuds capteurs déployés dans un réseau peut être de l'ordre de centaines voire de milliers et il peut atteindre quelques millions pour certaines applications. Les modèles de déploiement doivent être capables de travailler avec ce grand nombre de nœuds afin de garantir le bon fonctionnement du réseau.

I.3.6.3 Coût de production

Le coût d'un seul nœud capteur influence largement le coût global du réseau et les Réseaux de Capteurs consistent en un nombre important de nœuds. Le coût de chaque nœud capteur doit donc être réduit pour que l'utilisation de cette nouvelle technologie soit financièrement justifiée.

I.3.6.4 Environnement

Les nœuds capteurs doivent pouvoir travailler sans surveillance dans des zones géographiquement éloignées et inaccessibles quelque soit les différentes conditions d'environnement (l'intérieur de grosses machines, au fond d'un océan, dans un champ contaminé biologiquement ou chimiquement, sur un champ de bataille ou dans une maison).

I.3.6.5 Support de transmission

La communication entre les nœuds est normalement mise en œuvre en utilisant une communication radio sur les bandes ISM populaires. Cependant, certains RCSFs utilisent une communication optique ou infrarouge. Cette dernière ayant l'avantage d'être robuste et consomme moins d'énergie.

I.3.6.6 Consommation d'énergie

La taille des nœuds limite la taille de la batterie. En effet, la recharge des sources d'énergie est souvent coûteuse et parfois impossible. Ainsi, la durée de vie d'un nœud capteur est fortement liée à celle de la batterie. Par conséquent, la conception logicielle et matérielle doit prendre en compte les problèmes d'utilisation efficace de l'énergie car le dysfonctionnement d'un nœud implique des modifications topologiques et peut nécessiter le réacheminement des paquets et la réorganisation du réseau.

I.3.7 Impact de l'environnement de déploiement sur le Canal Sans Fil

Le canal sans fil est généralement dynamique et imprévisible et dépend principalement de son environnement, contrairement à un canal filaire qui est plutôt statique et prévisible, ce qui influence les performances des systèmes de communication sans fil et rend l'analyse exacte de ces systèmes souvent difficile. Dans les communications sans fil, la propagation radio fait référence au comportement des ondes radio lorsqu'elles se propagent d'un émetteur à un récepteur. Il existe une multitude de modèles de propagations radio qui peuvent estimer la puissance du signal dans un environnement précis [14].

I.3.7.1 Modèle de propagation en espace libre

Le modèle de propagation en espace libre est utilisé pour prévoir la puissance du signal reçu dans un environnement en visibilité directe (en Anglais, Line-Of-Sight -LOS-) sans obstacles entre l'émetteur et le récepteur séparés par une distance d . la puissance reçue notée $P_r(d)$ est exprimée par l'équation de Friis suivante :

$$P_r(d) = \frac{P_t \times G_t \times G_r \times \lambda^2}{(4\pi)^2 \times d^2 \times L} \quad (\text{I.1})$$

où G_e et G_r représentent les gains des antennes de transmission qui ont un rayonnement uniforme dans toutes les directions, P_t représente la puissance d'émission (en

watts), λ la longueur d'onde et L le facteur de perte du système [14].

I.3.7.2 Modèle Log-distance path loss

Une forme plus généralisée du modèle path loss peut être construite en modifiant le path loss (l'atténuation de parcours) en espace libre avec l'exposant de path-loss n qui varie en fonction des environnements. C'est ce que l'on appelle le modèle log-distance path loss qui est une extension du modèle d'espace libre Friis [14]. L'équation correspondante à ce modèle est la suivante :

$$PL(d)[dB] = PL(d_0) + 10 \times n \times \log_{10}\left(\frac{d}{d_0}\right) \quad (I.2)$$

- $PL(d)$ est l'atténuation de parcours à la distance d entre l'émetteur et le destinataire (d est mesurée en mètre).
- $PL(d_0)$ est l'atténuation de parcours à la distance de référence d_0 (d_0 est mesurée en mètre).
- d_0 est la distance de référence.
- n est l'exposant de l'atténuation de parcours.

Comme le montre le Tableau I.2, l'exposant d'atténuation de parcours peut varier de 2 à 6 en fonction de l'environnement de propagation.

| Environnement | Path loss exponent(n) |
|-------------------------------|-----------------------|
| Free space | 2 |
| Urban area cellular radio | 2.7-3.5 |
| Shadowed urban cellular radio | 3-5 |
| In building line-of-sight | 1.6-1.8 |
| Obstructed in building | 4-6 |
| Obstructed in factories | 2-3 |

Tableau I.2 – L'exposant de l'atténuation de parcours [47].

I.3.7.3 Modèle log-normal shadowing

Chaque chemin peut avoir un path loss différent même si la distance entre l'émetteur et le destinataire est égale, car les milieux environnants peuvent varier en fonction de l'emplacement du destinataire (la présence d'obstacles). Nous notons que les modèles précédents ne tiennent pas compte de cette situation particulière. Le modèle log-normal shadowing est utile lorsqu'il s'agit d'une situation plus réaliste [18]. Étant donné X_σ une variable aléatoire gaussienne de moyenne nulle et d'écart-type σ , et d_0 la distance de référence, le modèle log-normal shadowing est défini par l'équation suivante : [14] :

$$PL(d)[dB] = PL(d_0) + 10 \times n \times \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \quad (I.3)$$

I.3.7.4 Modèle Okumura-Hata

Le modèle Okumura-Hata est l'un des modèles de propagation les plus utilisés. Il a été développé grâce aux travaux de Okumura [45] qui a effectué de nombreuses expériences visant à calculer la hauteur d'antenne et la zone de couverture des systèmes de communication cellulaire dans les environs de Tokyo (Japon). Ce modèle a été étendu par Hata [28] pour couvrir divers environnements, y compris les zones urbaines, suburbaines et ouvertes, et est désormais appelé modèle Hata (ou Okumura-Hata) [14]. Il convient à un ensemble de paramètres qui sont montrés dans le Tableau I.3.

| | Minimum value | Maximum value |
|---------------------------------|---------------|---------------|
| Carrier frequency f_c (MHz) | 150 | 1500 |
| Base station height h_b (m) | 30 | 200 |
| Mobile station height h_m (m) | 1 | 10 |
| distance d (Km) | 1 | 20 |

Tableau I.3 – Plages des paramètres pour le Modèle Okumura-Hata [30] .

Le modèle Hata est formulé comme suite :

— **L'atténuation de parcours dans les zones urbaines**

$$PL_{Hata,U}(d)[dB] = 69.55 + 26.16 \log_{10} f_c - 13.82 \times \log_{10} h_B - CH + [44.9 - 6.55 \times \log_{10} h_B] \times \log_{10} \times d \quad (I.4)$$

Pour les couvertures de taille petite à moyenne :

$$CH = 0.8 + (1.1 \times \log_{10} \times f_c - 0.7) \times h_M - 1.56 \log_{10} \times f_c \quad (I.5)$$

Pour une couverture de grande taille :

$$CH = \begin{cases} 8.29 \times (\log_{10}(1.54 \times h_M)) \times 2 - 1.1 & \text{if } 150 \leq f_c \leq 200 \\ 3.2 \times (\log_{10}(11.7 \times h_M)) \times 2 - 4.97 & \text{if } 200 \leq f_c \leq 1500 \end{cases}$$

— h_B et h_M sont les hauteurs des antennes en mètre.

— f_c est la fréquence de transmission en Mégahertz (MHz).

— CH est le coefficient de corrélation de l'antenne de réception.

— d la distance en kilometer (km).

— L'atténuation de parcours dans les zones suburbaines :

$$PL_{Hata,SU}(d)[dB] = PL_{Hata,U}(d) - 2 \times \left(\log_{10} \frac{f_c}{28}\right)^2 - 5.4 \quad (I.6)$$

— L'atténuation de parcours dans les zones ouvertes :

$$PL_{Hata,O}(d)[dB] = PL_{Hata,SU}(d) - 4.78 \times (\log_{10} f_c)^2 + 18.33 \times \log_{10} f_c - 40.97 \quad (I.7)$$

Partie 2 : Stratégies de déploiement des Nœuds Capteurs dans les RCSFs

I.4 Introduction

Les Réseaux de Capteurs Sans Fil facilitent de nombreuses tâches difficiles à accomplir. Toutefois, ils font face à certains défis tel que le déploiement des nœuds capteurs qui constitue l'un des problèmes fondamentaux des RCSFs. En effet, le déploiement a un impact significatif sur d'autres fonctionnalités telles que la couverture, la connectivité, l'économie d'énergie et la durée de vie. Dans cette partie, nous présentons une analyse générale et détaillée des différentes techniques de déploiement existantes tout en soulignant les facteurs et les objectifs principaux influençant la phase de déploiement.

I.5 Les méthodes de déploiement

Le déploiement des nœuds capteurs dans une zone d'intérêt peut généralement être de manière planifiée ou aléatoire (voir Figure I.5). Le type de nœuds capteurs, l'application et l'environnement dans lequel les nœuds fonctionnent sont tous des facteurs qui influencent le choix du type de déploiement. Dans des situations où on n'a pas une connaissance préalable des champs comme sur un champ de bataille, le déploiement aléatoire peut être le seul choix. Cependant, le déploiement déterministe est souvent nécessaire lorsque les nœuds capteurs coûtent cher ou lorsque leur position a un impact assez important sur leur fonctionnement [58].

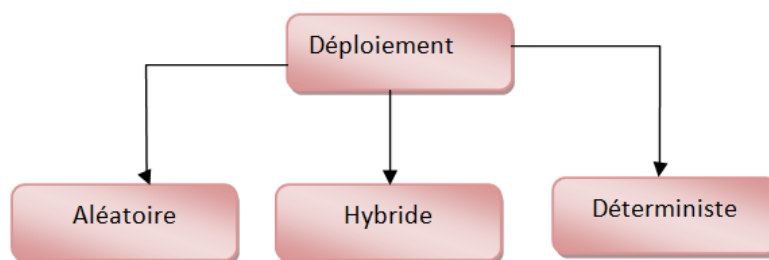


FIGURE I.5 – Les méthodes de déploiement.

I.5.1 Déploiement aléatoire

Cette méthode de placement est souvent utilisée pour déployer des nœuds capteurs dans des zones inconnues, difficiles d'accès ou inaccessibles. Les nœuds capteurs sont généralement dispersés par le biais d'un avion ou d'un drone, ce qui entraîne une distribution aléatoire des nœuds capteurs. Ce type de déploiement entraîne éventuellement des performances sous-optimales (par exemple, il ne garantit pas la couverture et la connectivité).

I.5.2 Déploiement déterministe

Ce type de déploiement est utilisé lorsque la zone d'intérêt est accessible. Dans ce cas, les nœuds capteurs sont placés dans des positions fixes et connues selon un plan prédéfini. Le placement planifié minimise le nombre de nœuds requis et maximise la couverture du champ de capture [58].

I.5.3 Déploiement hybride

Dans ce type de déploiement, les nœuds sont placés de manière non déterministe (aléatoire) dans des zones, matérialisant le champ de déploiement, créées de façon déterministe [26].

I.6 Stratégies de placement des nœuds

Les stratégies de placement des nœuds sont toutes déterministes et varient selon un ou plusieurs objectifs qu'on veut atteindre (maximiser la couverture, maximiser le nombre de voisins d'un nœud, etc.).

Ces méthodes sont classées en deux groupes en fonction de l'approche utilisée : approche basée sur les grilles et celle basée sur la géométrie algorithmique.

I.6.1 Stratégies basée sur les grilles

La stratégie basée sur la grille fournit un déploiement déterministe dans lequel la position des nœuds capteurs est fixée selon un modèle respectant une certaine distance, entre les nœuds, qui dépend de R_s (Rayon de capture) et de R_c (Rayon de communication), tel qu'un réseau triangulaire, une grille carré ou une grille hexagonale.

où chaque nœud capteur occupe une cellule hexagonal (voir Figure I.6), la couverture peut être contrôlé en ajustant la distance inter-nœuds d (la distance entre

deux nœuds capteurs adjacents), pour atteindre une couverture complète la distance inter-nœuds doit être $d \leq r\sqrt{3}$ où r est le rayon de détection.

I.6.1.1 Grille triangulaire

La Figure I.6 montre que chaque nœud a au maximum 6 voisins et au minimum 2. En terme de connectivité, ce type de stratégie assure jusqu'à 6-connectivité. Par contre en terme de couverture, ce modèle n'assure la couverture que si $d \leq \sqrt{3}R_s$ où d est la distance inter-nœuds (on obtient donc un triangle équilatéral) et R_s est le rayon de capture. On note que $R_c \geq d$ dont R_c est le rayon de communication.

Le nombre minimal de nœuds capteurs est calculé dans ce cas par $\frac{2\sqrt{3}}{9} \times \frac{1}{R_s^2}$ [53]. Le modèle triangulaire est le meilleur modèle car il est le moins coûteux parmi tous les types de stratégies basés sur la grille, en plus il offre la plus petite zone de chevauchement [4].

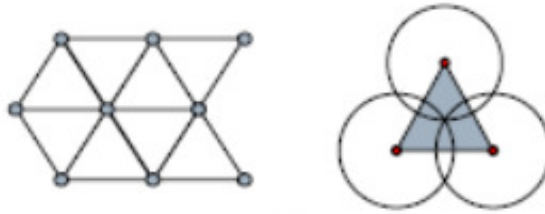


FIGURE I.6 – Modèle triangulaire [4].

I.6.1.2 Grille carré

Dans ce type de stratégie, la zone est divisée en cellules carrés, comme illustré dans la Figure I.7. Les nœuds sont situés soit sur les sommets de la cellule, soit au centre de la cellule [37]. Dans une architecture pareille, selon [53], une couverture complète et une connectivité sont garanties si la distance inter-nœuds est $d \leq \sqrt{2}R_s$ et $R_c \geq d$.

Le nombre minimal de nœuds capteurs est évalué dans ce cas à $\frac{1}{2} \times \frac{1}{R_s^2}$ qui est un peu coûteux par rapport au modèle triangulaire.

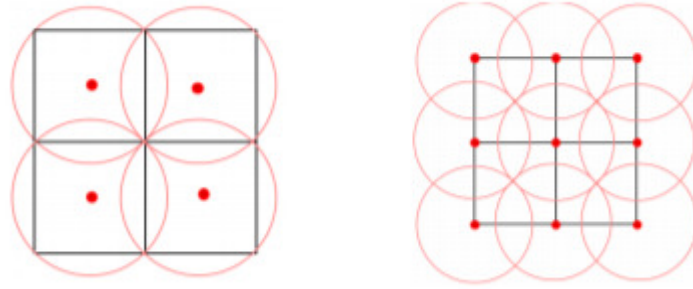


FIGURE I.7 – Modèle carré [37]

I.6.1.3 grille hexagonale

Ce type de stratégie garantit une couverture totale et une connectivité pour $d \leq R_s$ et $R_c \geq d$. Selon [53] le nombre minimal de nœuds capteurs est évalué dans ce cas à $\frac{4\sqrt{3}}{9} \times \frac{1}{R_s^2}$. Comme le montre la Figure I.8, les nœuds sont placés sur les sommets de l'hexagone. Ce modèle est considéré comme le modèle le plus coûteux par rapport aux deux modèles précédents. De surcroît, il donne les plus grandes zones de chevauchement [4].

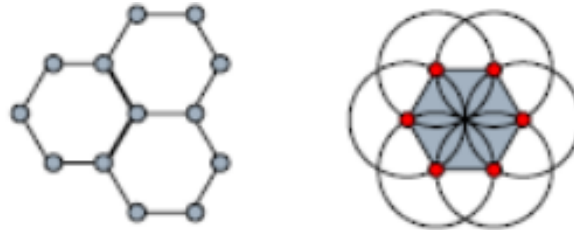


FIGURE I.8 – Modèle d'hexagone [4].

I.6.2 Les stratégies basées sur la géométrie algorithmique

Le domaine de la géométrie computationnelle est apparu dans les années 1970. Il traite de divers types de problèmes de calcul complexes de nature géométrique telles que la planification de mouvement dans le domaine de la robotique et la triangulation des polygones qui est utilisée pour résoudre le célèbre problème de la galerie d'art dans les applications de surveillance [17]. Le diagramme de Voronoï et la triangulation de Delaunay sont deux méthodes de la géométrie algorithmique utilisées dans les RCSFs pour résoudre les problèmes de trous de couverture en les détectant et en les éliminant afin de maximiser le taux de couverture.

I.6.2.1 Le diagramme de Voronoï

Le diagramme de Voronoï est une méthode de partitionnement de la zone en un certain nombre de polygones en fonction des distances par rapport à un ensemble discret spécifique de nœuds, comme illustré dans la Figure I.9. Chaque nœud n'occupe qu'un polygone et est plus proche de tout point de ce polygone que de tout autre nœud des polygones voisins. Ces polygones peuvent être obtenus en traçant le médiateur de chacun des deux nœuds voisins. Par conséquent, les arêtes des polygones sont à égale distance des nœuds voisins [17]. En raison de ces polygones de Voronoï, les nœuds peuvent déterminer des trous de couverture [16]. Ensuite, ils se déplacent afin de réduire ou d'éliminer ces trous tout en maximisant le taux de couverture de la zone considérée.

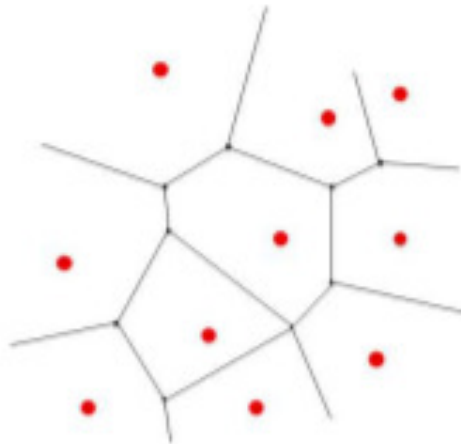


FIGURE I.9 – Le diagramme de Voronoï [37].

I.6.2.2 Triangulation de Delaunay

La triangulation de Delaunay est un type particulier de triangulation d'ensembles de points planaires. La triangulation de Delaunay d'un ensemble P est toujours une triangulation à angle optimal de P (Voir Figure I.10). La triangulation de Delaunay est étroitement liée au diagramme de Voronoï. Elle peut être construite en reliant chacun des deux points voisins du diagramme de Voronoï dont les polygones partagent un bord commun, comme le montre la Figure I.11 [17].

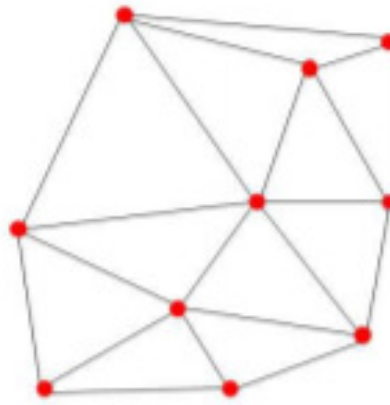


FIGURE I.10 – La Triangulation de Delaunay [37].

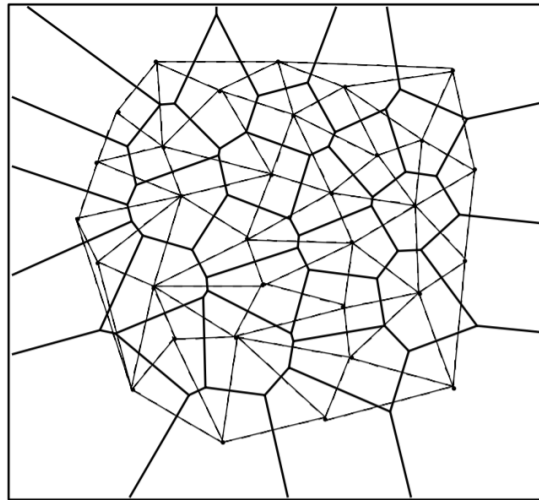


FIGURE I.11 – Delaunay-Voronoi [8]

I.7 Principaux objectifs du déploiement

I.7.1 Couverture

Parmi les facteurs importants du déploiement d'un réseau de capteurs, on citera la couverture qui est considérée comme l'une des métriques de la Qualité de Service (QoS). La couverture peut être complète ou partielle en fonction des besoins de l'application. Si n'importe quel point est surveillé par un seul nœud capteur (c'est-à-dire une couverture simple dite 1-couverture) ou par plusieurs nœuds capteurs (c'est-à-dire une k -couverture). La couverture dans les RCSFs peut être classée en trois types : couverture de zone, couverture de point (cible) et couverture de barrière [37].

I.7.1.1 La couverture de zone

Elle s'appelle aussi la couverture de surface. L'objectif principal est de surveiller une zone géographique appelée zone d'intérêt dans laquelle chaque point de cette région est couvert par un sous-ensemble de nœuds capteurs. Selon les besoins de l'application, une couverture totale ou partielle est requise.

I.7.1.2 Couverture de cible

Elle est dite aussi couverture de point. Ce type de couverture sert à surveiller uniquement certains points spécifiques (cibles) dans un champ de capture, dont la position géographique est connue et chaque point spécifique doit être couvert par au moins un nœud capteur. Parmi les exemples de surveillance de points d'intérêt, on peut citer les applications militaires telles que la surveillance des bases ennemies.

I.7.1.3 Couverture de barrière

Il s'agit de la Couverture de frontière afin de couvrir une partie de la zone d'intérêt. Les nœuds capteurs ne sont pas conçus pour surveiller les événements à l'intérieur de la zone considérée. En effet, on a besoin seulement de couvrir le périmètre de cette région pour détecter les intrus qui tentent de pénétrer dans cette zone. Ce type de couverture est connu pour être un modèle de couverture approprié pour des applications telles que la surveillance des frontières internationales ou la détection de propagation de produits chimiques mortels autour d'une usine, par exemple [55].

I.7.2 Connectivité

Deux nœuds capteurs sont dits connectés si et seulement ils peuvent communiquer directement (connectivité à un saut) ou indirectement (connectivité à sauts multiples). Dans les RCSFs, le réseau est considéré comme connecté s'il existe au moins un chemin entre le sink et chaque nœud capteur dans la zone considérée [37].

Il faut tenir compte de la connectivité pour bien surveiller une région donnée. Il ne suffit pas d'assurer une couverture seulement. Les nœuds capteurs doivent être capables de signaler immédiatement au sink tout événement détecté. Il existe deux types de connectivité réseau : connectivité complète et connectivité intermittente. La connectivité réseau complète peut également être simple (1-connectivité) : s'il y a un seul chemin depuis tout nœud capteur jusqu'au sink, ou multiple (k-connectivité) : si plusieurs chemins disjoints existent entre un nœud quelconque et le sink.

I.7.3 Latence et nombre de sauts

La latence est une métrique globale de la performance du réseau. Elle représente le temps nécessaire pour qu'un nœud transmette un paquet de données jusqu'au sink. Ce temps, exprimé en nombre de slots, comporte le temps d'attente nécessaire avant que le nœud émetteur puisse envoyer le paquet et le temps nécessaire pour que ce paquet arrive au sink. Notons que la latence dépend de la profondeur d'un nœud dans l'arbre de routage [41].

Le nombre de sauts est un critère de conception et de performance important dans les RCSFs. Il est défini comme étant le nombre de nœuds intermédiaires par lesquels les paquets doivent passer entre le nœud source et le nœud destination [34].

Enfin, nous tenons à souligner qu'un chemin avec un nombre minimal de sauts ne garantit pas une latence optimale ainsi qu'un chemin plus rapide (avec une faible latence) n'implique pas un chemin avec un nombre de sauts minimal [41].

I.7.4 Energie

L'une des contraintes importantes dans les RCSFs est l'exigence d'un fonctionnement avec une faible consommation d'énergie, car chaque nœud est alimenté par une batterie limitée. En général, un nœud capteur utilise son énergie en trois opérations principales : la capture, la communication (émission et réception) et le traitement des données [53].

- **La capture** : L'énergie consommée lors de la phase de capture varie en fonction du phénomène observé. Une capture sporadique consomme moins d'énergie qu'une détection constante.
- **La communication** : L'énergie consommée dans cette phase est la plus grande parmi les trois phases de consommation. Cette consommation est définie par la taille des données à communiquer et la distance de transmission, ainsi que par la puissance du signal. Quand la puissance d'émission est élevée, le signal aura une grande portée et l'énergie consommée sera plus élevée.
- **Traitement de données** : Cette phase consomme moins d'énergie que la communication. D'ailleurs, l'énergie nécessaire pour transmettre 1 KB sur une portée de 100 m est approximativement égal à celui nécessaire pour exécuter 3 millions d'instructions à une vitesse de 100 millions d'instructions par seconde [36]. Donc, parfois il est préférable de traiter les données localement surtout s'ils sont de type scalaire (température, humidité).

I.8 Etat de l'art sur le déploiement des Nœuds Capteurs dans les RCSFs

Dans [49], les auteurs ont mis au point des méthodes d'optimisation du déploiement des nœuds capteurs afin de trouver les différents emplacements permettant de déployer un nombre minimum de nœuds de capteurs assurant une couverture complète de la zone ainsi qu'une connectivité totale du réseau. Les auteurs ont développé un modèle de programmation linéaire en nombre entier pour résoudre le problème de manière optimale. Ils ont vérifié par des expériences de calcul, en utilisant le solveur Cplex, que le modèle ILP proposé est en mesure de fournir une solution optimale aux problèmes de petite et moyenne taille. Ils ont proposé également un algorithme de recherche locale (LS) et un algorithme génétique (GA) comme méthodes de résolution approchées. Ils ont également observé que leurs méthodes donnaient de meilleurs résultats en termes de nombre de nœuds requis que les méthodes de placement des nœuds capteurs classiques, telle que les méthodes utilisant les motifs hexagonales, carrés et triangles.

Les auteurs dans [21] ont considéré le problème de MCC : Minimum Connected Coverage ou ils ont visé à garantir que les nœuds capteurs doivent être placés de manière à couvrir toute la zone d'intérêt et que chaque nœud capteur peut transmettre ses données à la station de base via d'autres nœuds capteurs. Ils ont proposé deux formulations de programmation mathématique pour le problème MCC sur des graphiques à grille carré, puis, ils mes ont comparées avec le modèle proposé dans [49]. Les deux formulations ont été expérimentées pour différentes instances. Ils ont surpassées le modèle proposé ans [49], avec une proportion plus élevée d'instances résolues, un temps de calcul de processeur réduit et un nombre moins élevé de nœuds parcourus dans la recherche arborescente. Selon leurs expériences de calcul, les auteurs ont confirmé la difficulté de résoudre le problème du MCC en utilisant des formulations linéaires en nombres entiers mixtes, particulièrement pour les petites valeurs de Rayon de couverture et de communication.

Le travail présenté dans [19] a proposé un algorithme dans lequel une zone à surveiller est sous la forme d'un champ 2D délimité avec des points critiques prédéterminés dans le champ à surveiller par les nœuds capteurs. La technique proposée par les auteurs spécifie que les nœuds capteurs et relais ne peuvent pas être placés sur le point critique lui-même. Seule une zone spécifique est entièrement couverte car cette technique a pour objectif de réduire les coûts de déploiement. Ils ont formalisé le problème sous forme d'un programme linéaire en nombres entiers (ILP) pour déterminer la combinaison de coût minimum de nœuds capteurs et relais avec

leurs emplacements, de sorte que chaque point critique soit couvert. La solution obtenue garantit également la connectivité de chaque nœud capteur au Sink via des nœuds relais installés principalement pour cette raison. Finalement, une expérience a été menée pour observer la durée d'exécution du modèle pour un champ bidimensionnel de 10×10 en appliquant un nombre différent de points critiques (10, 15 et 20) générés de manière aléatoire dans le champ. Les auteurs ont noté que la durée d'exécution du programme linéaire en nombres entiers est petite pour des champs relativement petits, mais elle commence à augmenter de façon exponentielle à mesure que les dimensions du champ se développent et que des méthodes heuristiques sont nécessaires pour obtenir une solution approchée pour les grands espaces dans un temps de calcul relativement réduit.

Dans [51], les auteurs ont abordé le problème de placement des nœuds relais pour établir une connectivité complète dans un réseau. Le problème est modélisé comme un problème d'optimisation et résolu en utilisant des algorithmes bio-inspirés, à savoir l'algorithme MFO (Moth Flame Optimizer), l'algorithme ISA (Interior Search Algorithm) et l'algorithme BA (Bat Algorithm) pour identifier les positions des nœuds relais –RN-. Ils ont considéré deux types de nœuds : nœuds capteurs (SN) et nœuds relais (RN). Les SN surveillent en permanence l'environnement pour détecter et collecter les données. Les RN ont comme objectif de relayer les données collectées par les SN vers le Sink. Ils ont supposé que les nœuds capteurs sont déployés de manière aléatoire et que les emplacements de déploiement sont définis par la station de base. Les nœuds relais sont déployés au cas où un ou plusieurs nœuds capteurs ne pourraient pas communiquer avec le Sink par le biais de nœuds capteurs intermédiaires. Les algorithmes d'optimisation proposés mettent à jour les positions des nœuds relais de manière itérative, pour obtenir la meilleure solution globale possible. La solution globale consiste en un ensemble minimal de nœuds relais, placés de manière optimale, de sorte que le réseau soit entièrement connecté.

I.9 Conclusion

Les Réseaux de Capteurs sans Fil sont un domaine de recherche en évolution continue avec une multitude de contextes d'application.

Dans la première partie de ce Chapitre, nous avons introduit les RCSFs en décrivant leur architecture, leurs caractéristiques, les facteurs influençant leur conception, ainsi que leurs domaines d'application.

Dans la deuxième partie, nous avons mis en évidence les différentes stratégies de déploiement des RCSFs ainsi que l'influence du positionnement déterministe des

noeuds capteurs sur les performances de ces réseaux, notamment en termes de couverture, de connectivité, de coût, et de durée de vie du réseau. Dans le prochain Chapitre, nous allons présenter un état de l'art sur les techniques d'optimisation combinatoire, mono et multi-objectifs qui tentent d'améliorer les critères de performance d'un RCSF, à travers un positionnement adéquat.

Optimisation Combinatoire : Etat de l'art

II.1 Introduction

L'optimisation combinatoire est une discipline mathématique qui vise à utiliser des techniques combinatoires pour résoudre des problèmes d'optimisation discrète. La résolution de tels problèmes consiste à déterminer la meilleure solution possible à partir d'un ensemble fini de possibilités. L'optimisation combinatoire est utilisée dans de nombreux domaines ayant besoin d'outils d'aide à la décision, tels que la conception des réseaux de télécommunication, le placement et l'ordonnancement de tâches sur les processeurs, la bio-informatique, et aussi pour répondre aux besoins croissants des secteurs économiques et industriels (maximisation des performances et minimisation des coûts). Dans ce chapitre, nous allons présenter les concepts fondamentaux de l'optimisation combinatoire.

II.2 Problèmes d'optimisation combinatoire

II.2.1 Définition

Un problème d'optimisation combinatoire est défini par :

- **Un espace de recherche** : souvent cet espace est fini et très grand. Il représente les diverses valeurs prises par les variables de décision.
- **Une fonction objectif** : c'est le but à atteindre et à optimiser. Elle peut concerner un (problème mono-objectif) ou plusieurs critères (problème multi-objectif).
- **Un ensemble de contraintes** : il s'agit des restrictions imposées qui doivent être respectées et permettent de se limiter aux solutions réalisables.

La plupart de ces problèmes sont dits des problèmes NP-dur car en fonction de la taille du problème et le nombre d'objectifs à optimiser il n'existe pas des algorithmes qui offrent une solution exacte en un temps polynomial.

La classe NP (Non-déterministe Polynomial) : représente les problèmes dont les solutions peuvent être vérifiées en temps polynomial.

Problème NP-complet : est un problème de décision qui appartient à la classe NP et dont le meilleur algorithme de résolution connu à ce jour est exponentiel en la taille du problème dans le pire des cas.

Cette classe représente les problèmes qui s'expriment sous la forme d'une question mathématique dont la réponse ne peut être que « oui » ou « non ».

Par exemple, le test de primalité est un problème de décision qui consiste à répondre à la question suivante : « étant donné un entier n , est-ce que n est premier ? ». De même le problème de coloration de graphe peut s'exprimer sous la forme d'un problème de décision « étant donné un graphe G et un entier k , est-ce que G est k -coloriable ? c'est-à-dire est-il possible de colorier tous les sommets de G en utilisant uniquement k couleurs ? ».

Notons que le deuxième est un cas classique de problèmes NP-complets alors que le premier ne fait pas partie de cette classe.

Problème NP-dure est un problème dont la solution ne peut être vérifiée en un temps polynômial et tout problème NP-complet peut être réduit polynômialement à ce problème. Les problèmes NP-dure sont donc des problèmes au moins aussi difficiles que les problèmes NP-complets. En général, on utilise cette classe pour les problèmes d'optimisation, pour lesquels on ne peut pas vérifier la solution en temps polynomial. Exemples de problèmes NP-dure : le problème du voyageur de commerce, le nombre chromatique d'un graphe. . .

II.2.2 Quelques problèmes classiques de l'optimisation combinatoire

II.2.2.1 Problème du sac à dos (Knapsack Problem)

Le problème du sac à dos est l'un des problèmes difficiles de l'optimisation combinatoire puisque il appartient à l'ensemble des problèmes NP-dur.

Soit un sac de capacité C et un ensemble E de n objets $E = \{o_1, o_2, \dots, o_i, \dots, o_n\}$. Chaque objet o_i a un poids p_i et un bénéfice b_i . L'objectif est de trouver un ensemble d'objets appartenant à E de telle sorte qu'on maximise le bénéfice sans avoir une charge supérieure à la capacité C du sac à dos [24]. Ce problème se formalise comme suite :

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^n b_i \cdot x_i \\ & \text{subject to} && \sum_{i=1}^n p_i \cdot x_i \leq C \\ & && x_i \in \{0, 1\}, 1 \leq i \leq n \end{aligned}$$

II.2.2.2 Problème du voyageur de commerce (Travelling Salesman Problem -TSP-)

Soit un graphe $G=(S,A)$ où S est l'ensemble des villes représentées par les sommets et A l'ensemble des arcs representes les lien entre les villes avec une certaine distance d . On cherche le plus court chemin en termes de distance, de telle sorte qu'on visite chaque ville exactement une seule fois et on revient à la ville du départ, comme illustré dans la Figure II.1 (Cycle Hamiltonien). Ce problème appartient à l'ensemble des problèmes NP-dur [50].

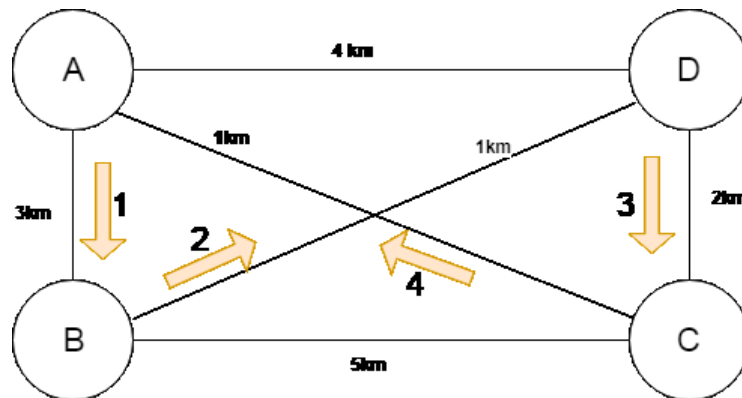


FIGURE II.1 – Exemple de cycle Hamiltonien

II.2.2.3 Problème de coloration de graphe

La coloration de graphe consiste à colorier chaque sommet avec une couleur en respectant le fait que chaque deux sommets adjacents n'ont pas la même couleur. L'objectif est de minimiser le nombre de couleurs utilisées qui s'appelle en théorie des graphes le nombre chromatique. La Figure II.2 représente un exemple de coloration de graphe. Les domaines d'application de la coloration de graphe est vaste incluant le problème de l'attribution de fréquences dans les réseaux de télécommunication, etc [40].

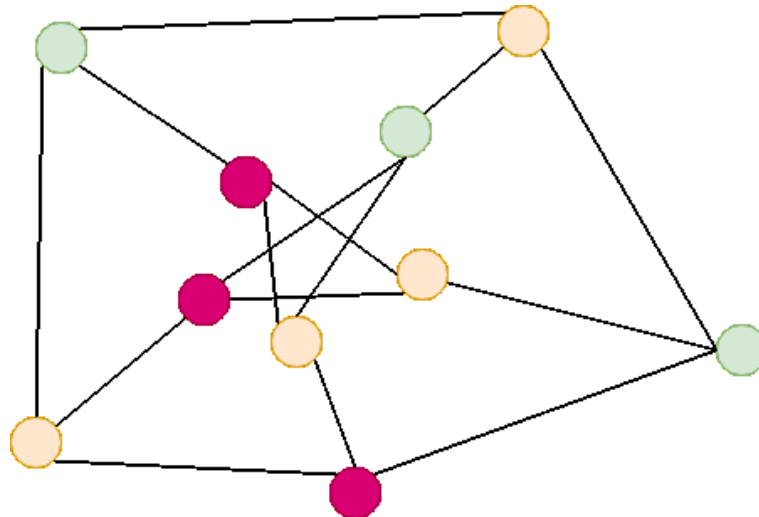


FIGURE II.2 – Exemple de coloration de graphe

II.3 L'optimisation combinatoire mono-objectif

L'optimisation combinatoire mono-objectif est basée sur la minimisation ou la maximisation, d'une seule fonction objectif f sur l'ensemble des solutions réalisables \mathcal{D} , pour trouver la solution optimale qui est déterminée suivant un seul critère du problème donné. Plus formellement, on note un tel problème [52] :

$$\min_{x \in \mathcal{D}} f(x) \text{ ou } \max_{x \in \mathcal{D}} f(x)$$

- On appelle tout point $u \in \mathcal{D}$, tel que $\forall x \in \mathcal{D}, f(u) \leq f(x)$, un minimum (minimum global).
- On appelle tout point $u \in \mathcal{D}$, tel que $\forall x \in \mathcal{D}, f(u) \geq f(x)$, un maximum (maximum global).
- la valeur $f(u)$ représente le minimum (ou le maximum) et est notée f_{\min} (ou f_{\max}).
- L'ensemble \mathcal{D} est appelé le domaine admissible (ensemble des solutions réalisables), et la fonction f fonction objectif.

II.4 L'optimisation combinatoire multi-objectif

L'optimisation combinatoire multi-objectif se distingue de l'optimisation combinatoire mono-objectif par le nombre de fonctions objectifs à optimiser simultanément qui sont souvent contradictoires. La solution d'un PMO (Problème Multi-Objectif) n'est pas unique, mais un ensemble de solutions appelé *Pareto optimales* noté PO

[54].

Un PMO peut être défini de la manière suivante :

$$(PMO) \begin{cases} \text{Optimiser } f(x) = (f_1(x), f_2(x), \dots, f_n(x)) \\ \text{sous } x \in D \end{cases}$$

où n est le nombre d'objectifs ($n \geq 2$), $x = (x_1, x_2, \dots, x_k)$ est le vecteur représentant les variables de décision, D représente l'ensemble des solutions réalisables et chacune des fonctions $f_i(x)$ est à optimiser, c'est-à-dire à minimiser ou à maximiser.

Après avoir obtenu l'ensemble des solutions Pareto, le choix d'une seule solution parmi cet ensemble revient au décideur. Les méthodes de résolution de problèmes multi-objectif sont donc des méthodes d'aide à la décision et le choix final sera laissé au décideur. Ces méthodes peuvent être réparties en trois familles en se basant sur le type de coopération solveur-décideur : [54].

1. A priori

Dans cette approche le problème est transformé en un problème mono-objectif et résolu. Ce Procédé consiste à combiner les différentes fonctions objectifs en une seule fonction. Dans ce cas, le compromis que l'on désire faire entre les objectifs à été défini avant l'exécution de la méthode.

Preferences → Research

2. A posteriori

Dans cette approche, on cherche à générer un ensemble de bonnes solutions au décideur en lançant l'algorithme d'optimisation en premier, puis il choisit parmi celles-ci la solution qui lui convient le plus.

Research → Preferences

Cette approche est utilisable dans le cas où la cardinalité de l'ensemble PO est réduite.

3. Interactive

Cette approche prend en compte les préférences formulées à partir des connaissances a priori du décideur et celles acquises pendant la résolution du problème, c'est-à-dire une coopération progressive entre le solveur et le décideur est appliquée, comme le montre la Figure II.3 .

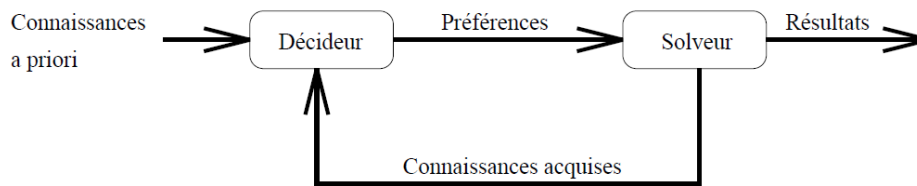


FIGURE II.3 – Approche Interactive [54].

II.4.1 Stratégies de prise en compte des objectifs

Comme l'optimisation combinatoire multi-critères (multi-objectif) traite un certain nombre d'objectifs, il existe plusieurs méthodes pour prendre en considération ces objectifs [54] :

II.4.1.1 Les méthodes non agrégées et non pareto

Dans ces méthodes les objectifs peuvent être traités selon un principe de “selection parallèle”. L'ensemble de solutions initial est divisé en un nombre de sous-ensembles et chaque sous-ensemble i est sélectionné suivant l'objectif f_i , donc on obtient un nombre de sous-ensembles égal au nombre d'objectifs.

La méthode lexicographique est une autre approche non-pareto qui sélectionne et traite les objectifs selon un ordre défini par le décideur. Cet ordre représente l'importance des objectifs. Les fonctions sont résolues l'une après l'autre en utilisant la solution précédente jusqu'à obtenir la dernière solution qui représente l'optimum global.

Nous tenons à souligner que nous avons utilisé cette méthode pour la formalisation de notre problème de déploiement des nœuds du RCSF.

II.4.1.2 Les méthodes agrégées

Ces méthodes sont basées sur la transformation du PMO en un problème uni-objectif (mono-objectif), telle que la méthode d'agrégation qui combine les fonctions objectif en une seule fonction, la méthode E-contrainte qui consiste à optimiser une seule fonction objectif (primaire ou préférée) tandis que les autres objectifs sont traités comme des contraintes. Une autre méthode agrégée est la programmation par but où le décideur définit ses références ou buts pour chaque objectif.

II.4.1.3 Les méthode Pareto

Ces methodes utilisent la notion de dominance et consistent à trouver l'ensemble du Front Pareto. Ce dernier représente l'ensemble des solutions efficaces (ou non dominées) où aucune fonction objectif ne peut être améliorée sans dégrader certaines des autres fonctions.

Definition II.4.1. *Pareto dominance*

On dit que le vecteur $x \in D$ domine le vecteur $y \in D$ ssi :

$$f_i(x) \leq f_i(y), \forall i \in \{1, 2, \dots, n\}, \exists j \in \{1, 2, \dots, n\} / f_j(x) < f_j(y)$$

On note $x \preceq y$ et on lit : x Pareto domine y .

II.5 Approches classiques de résolution des problèmes d'optimisation combinatoire

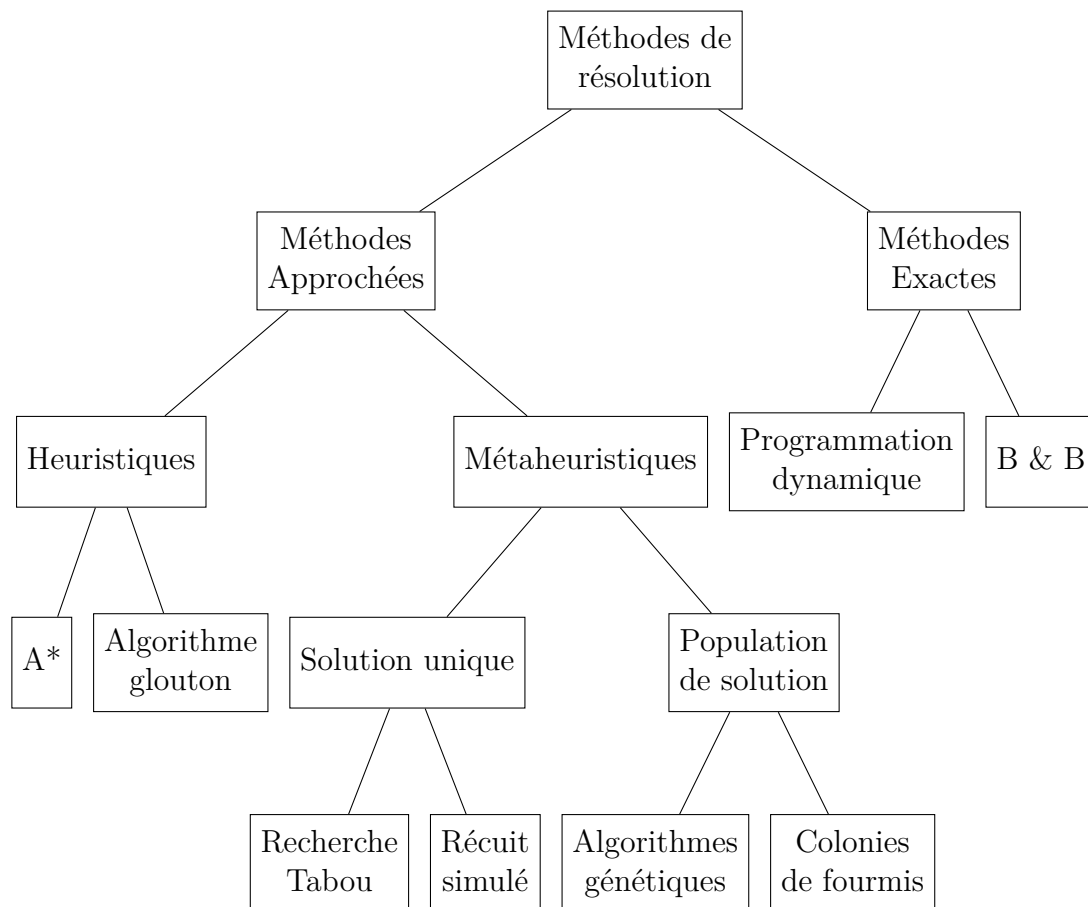


FIGURE II.4 – Classification des méthodes de résolution

II.5.1 Les méthodes exactes

Les méthodes exactes garantissent la complétude de la résolution, c'est le cas de Branch and Bound et de la programmation dynamique. De telles méthodes sont généralement adaptées pour aborder des problèmes de petite ou de moyenne taille pour pouvoir explorer l'espace des solutions en un temps raisonnable. Autrement, le temps de calcul nécessaire à la résolution augmente exponentiellement avec la taille du problème à résoudre (dans le pire des cas).

II.5.1.1 Branch and Bound

La méthode d'évaluation et de séparation (B & B) est une méthode de résolution de problèmes d'optimisation combinatoire NP-dur, en particulier, elle permet la résolution des problèmes de programmation linéaire en nombres entiers. Elle utilise la stratégie diviser pour régner et son principe est de découper le problème en sous-problèmes pour les optimiser chacun individuellement. Cette méthode est basée sur une recherche arborescente d'une solution optimale dont la racine contient l'ensemble de toutes les solutions réalisables [12].

Le branch-and-bound est basé sur trois axes principaux :

1. Le principe de séparation : Consiste à diviser en fonction d'un certain critère le problème P en sous problèmes P_i qui ont chacun son ensemble de solutions réalisables contenues dans un sommet de l'arborescence, comme le montre la Figure II.5. L'union des sous-ensembles associés aux fils d'un sommet doit être égale à l'ensemble associé à ce sommet. La solution optimale est obtenue en calculant la valeur de la fonction objectif pour toutes les feuilles non vides de l'arborescence.
2. L'évaluation : Elle permet de définir l'optimum pour le sous-ensemble courant lorsque le sous-problème devient simple et on peut le résoudre directement. Cette méthode permet aussi d'éviter de séparer certains sommets de l'arborescence dans le cas où il est prouvé que l'ensemble associé au nœud en question ne contient pas de solutions candidates à l'optimalité. Cela est possible grâce à la connaissance d'une borne inférieure (respectivement une borne supérieure) de chaque sous-problème. Si on arrive à trouver un minorant (dans le cas d'un problème de minimisation) supérieur à la meilleure solution trouvée jusqu'à présent, on peut dire que le sous-ensemble ne contient pas l'optimum.
3. Stratégie de parcours :

Pour choisir le prochain sommet à séparer parmi l'ensemble des sommets de l'arborescence, il existe plusieurs stratégies de parcours. On peut citer :

- La profondeur d'abord : Cette stratégie avantage les sommets les plus éloignés de la racine en appliquant plus de séparations au problème initial.
- La largeur d'abord : On favorise les sous-problèmes obtenus par le moins de séparations du problème de départ (les sommets les plus proches).
- Le meilleur d'abord : Elle consiste à explorer les sous-problèmes là où la probabilité de trouver une meilleure solution est la plus grande.

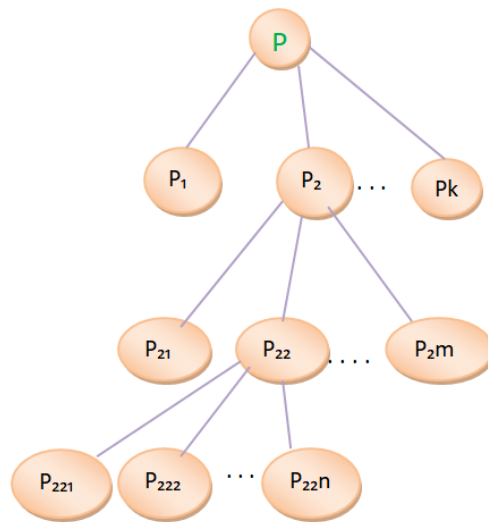


FIGURE II.5 – Division D'un Problème en sous-Problèmes

La méthode de coupes

Cette méthode consiste à résoudre un problème relaxé du problème d'optimisation combinatoire P et à ajouter itérativement des contraintes du problème initial. A partir du problème relaxé, nous obtenons une solution. Si cette solution viole certaines contraintes, on dit qu'elles sont des coupes planes. On rajoute alors une coupe plane au problème relaxé, et on regarde si cela engendre de nouveau des coupes planes. Quand l'algorithme s'arrête, il n'y a plus de contraintes violées par la solution courante, donc elle constitue une solution optimale du problème initial.

Branch and Cut

Branch-and-Cut utilise deux techniques : le Branch-and-Bound et la méthode de coupes (Cutting planes method) présentés précédemment.

Le "Branch and Cut" commence par résoudre une relaxation du problème puis il applique la méthode des coupes sur la solution trouvée. Si celle-ci n'arrive pas à

obtenir une solution entière alors le problème est divisé en plusieurs sous-problèmes qui seront résolus de la même façon [9].

En fin, nous tenons à souligner qu'il existe d'autres améliorations de l'algorithme B & B, à savoir les algorithmes : B & P (Branch and Price) [5], B& C & P (Branch and Cut and Price) [42].

II.5.1.2 Programmation Dynamique

Richard Bellman a initié la programmation dynamique [6]. Elle est basée sur le principe d'optimalité de Bellman : *Une solution d'un problème s'obtient en combinant des solutions optimales à des sous-problèmes.*

Cette méthode partitionne le problème initial en sous-problèmes indépendants et les résout récursivement, la solution optimale est la combinaison des solutions des sous-ensembles. Les solutions de l'ensemble sont déduites progressivement en débutant par les solutions des sous-problèmes les plus petits, donc les solutions sont calculées de manière ascendante.

La programmation dynamique est connue pour résoudre des problèmes répandus comme le problème du sac à dos, la tour de Hanoi et le plus court chemin par Dijkstra [32].

Algorithme de Floyd-Warshall

L'algorithme de Floyd-Warshall [23] est un exemple de programmation dynamique [15]. IL s'agit d'un algorithme simple et largement utilisé pour calculer les plus courts chemins entre toutes les paires de sommets dans un graphe orienté et pondéré.

Les poids des arcs du graphe d'entrée peuvent être négatifs, cependant le graphe ne doit pas contenir un circuit négatif pour que l'algorithme génère un résultat correct. L'algorithme de Floyd-Warshall ne peut pas résoudre le problème du plus court chemin efficacement tant qu'il existe un circuit négatif mais il peut détecter la présence d'un circuit négatif en vérifiant qu'il existe une entrée négative dans la diagonale de la matrice de sortie *dist* (lignes 11,12 et 13 de L'Algorithme 1) [31].

Algorithm 1 Floyd-Warshall Algorithm

```

1: Let  $dist$  be a  $|V| \times |V|$  matrix of minimum distances initialized to  $\infty \forall i \neq j$ 
2: for each vertex  $v$  do
3:    $dist[v][v] \leftarrow 0$ 
4: for each edge  $(u, v)$  do
5:    $dist[u][v] \leftarrow w(u, v)$  ▷ the weight of the edge  $(u, v)$ 
6: for  $i$  from 1 to  $|V|$  do
7:   for  $j$  from 1 to  $|V|$  do
8:     for  $k$  from 1 to  $|V|$  do
9:       if  $dist[j][k] > dist[j][i] + dist[i][k]$  then
10:         $dist[j][k] \leftarrow dist[j][i] + dist[i][k]$ 
11: for  $i$  from 1 to  $|V|$  do
12:   if  $dist[i][i] < 0$  then
13:     return ('graph contains a negative cycle')

```

II.5.2 Les méthodes Approchées

II.5.2.1 Heuristiques

Les heuristiques, appelées aussi méthodes approximatives car elles fournissent des solutions bonnes et possibles en un temps raisonnable ou polynomial mais ne sont pas forcément optimales. Ces techniques sont conçues pour résoudre des problèmes spécifiques et elles ne parcourent pas tout l'espace de recherche (ne sont pas des méthodes complètes) contrairement aux méthodes exactes. Parmi ces heuristiques, on peut citer les algorithmes glouton et l'algorithme A^* .

— **Algorithme glouton**

Un algorithme glouton (Greedy Algorithm) est un algorithme qui permet de résoudre des problèmes d'optimisation combinatoire passant généralement par une séquence d'étapes, avec un ensemble de choix à chaque étape. Il fait un choix localement optimal (on résout le sous-problème) dans l'espoir que ce choix aboutira à une solution globalement optimale. Les algorithmes gloutons ne donnent pas toujours des solutions optimales, mais ils le font pour de nombreux problèmes tels que les algorithmes de rendu de monnaie, les problèmes de coloration de graphes.

— **Algorithme A^***

L'algorithme A^* a été présenté à l'origine par Peter E. Hart, Nils John Nilsson et Bertram Raphael en 1968 [27]. Il a été conçu pour résoudre le problème du plus court chemin dans un graphe entre un nœud source et un nœud destination. Il s'agit d'une extension de l'algorithme de Dijkstra. Le but principal de cet algo-

rithme est que la première solution fournie soit l'une des meilleures.

Soit un graph donné, à partir d'un nœud source l'algorithme choisit le nœud qui minimise le coût qui est défini préalablement jusqu'à arriver au nœud destination. Ce principe est appelé la recherche du premier meilleur nœud ("Best First Search"). A^* utilise une fonction d'estimation de coût pour le choix du prochain nœud à visiter $d(i)$, tel que :

$$d(i) = g(i) + h(i) \tag{II.1}$$

Où :

$d(i)$: le coût total estimé du chemin.

$g(i)$: est la longueur estimée du plus court chemin de l'origine s au nœud i .

$h(i)$: est la longueur estimée du plus court chemin de i à la destination t .

Si $h(i) = 0$ pour tous les nœuds, alors A^* est identique à l'algorithme de Dijkstra [59].

II.5.2.2 Méta-heuristiques

Contrairement aux heuristiques qui sont spécifiques à certains problèmes, les méta-heuristiques sont conçues pour résoudre des problèmes d'optimisation combinatoire difficiles généraux de tous types (mono-objectif, multi-objectif). Elles sont donc totalement indépendantes du problème abordé. Elles ne garantissent pas de trouver de manière exacte l'ensemble PO, mais une approximation de cet ensemble notée PO^* [54]. Les méta-heuristiques sont basées sur l'amélioration itérative d'une population de solutions ou d'une solution unique (par exemple la recherche Tabou) ainsi qu'elles sont inspirées par des phénomènes physiques (par exemple le recuit simulé) ou de la nature et la biologie (Algorithme Génétique, Algorithme de Colonie de Fourmis) [52].

— Recuit simulé

L'algorithme d'optimisation combinatoire recuit simulé (Simulated Annealing) a été établi indépendamment par les trois chercheurs de la société IBM, S. Kirkpatrick, C.D. Gelatt et M.P. Vecchi en 1983 [38], et par Cerny en 1985 [13]. L'origine de cette méthode vient de l'analogie avec la métallurgie, où un processus appelé "recuit" consiste à partir d'une haute température du solide, puis à le laisser refroidir lentement pour atteindre des états de basse énergie. Le but est d'atteindre un état de quasi équilibre thermodynamique. Cet état est la solution du problème.

La méthode est appliquée aux problèmes d'optimisation combinatoire . Elle commence par générer une solution initiale s à partir de l'ensemble de recherche aléatoirement dans le but de minimiser une fonction objectif f ou $f(s)$ est l'énergie. Ensuite, elle détermine un paramètre T qui décroît pour tendre vers 0. A chaque itération, on perturbe la solution pour entraîner une variation Δ de l'énergie du système. Si cette variation fait baisser l'énergie du système, elle est appliquée à la solution courante. Sinon, la solution sera acceptée avec une probabilité $\exp \frac{\Delta}{T}$. La théorie démontre que le recuit simulé donne une solution approchée à la solution optimale du problème plus rapidement qu'une exploration exhaustive. Pratiquement, il faut utiliser les bons paramètres internes de l'algorithme pour accélérer la convergence vers une solution pseudo-optimale [32].

— Recherche Tabou

L'algorithme de Recherche tabou (Tabu Search) a été proposé par Glover en 1986 [25]. Il a connu un grand succès dans les années 90 et a montré sa performance sur de nombreux problèmes d'optimisation combinatoire.

A nos jours, il garde sa place dans l'ensemble des métaheuristiques à solution unique les plus utilisées.

La particularité de cette méthode est l'utilisation d'une mémoire pour enregistrer les informations associées au processus de recherche de façon à rendre la recherche un peu intelligente.

Donc l'idée est de passer d'une solution courante X à une autre solution Y telle que Y est une solution voisine de X . Dans le cas où toutes les solutions voisines ne sont pas améliorantes, la méthode accepte la solution la moins bonne que la solution courante pour éviter des optimums locaux. Habituellement, tous les voisins sont exploités de manière déterministe comme dans une recherche locale, le meilleur voisin remplace la solution courante.

Quand on est confronté à un optimum local, la recherche continue en choisissant le candidat le moins mauvais que la solution courante. Le meilleur voisin est considéré comme nouvelle solution même s'il n'améliore pas la solution courante, donc on peut revenir à des solutions déjà vérifiées, ce qu'on appelle un cycle.

Pour éviter ce phénomène, la méthode gère une mémoire des solutions récemment visitées appelée Liste Tabou. Elle contiendra tous les mouvements que l'on a déjà effectué et qui deviennent donc interdits (tabous). Ceci nous permet d'éviter de tomber dans un cycle de mouvements répétitifs et de sortir ainsi des minima locaux [33].

— **Algorithmes génétiques**

Les algorithmes génétiques (Genetic algorithms) font partie des algorithmes évolutionnaires. L'objectif de ces algorithmes est de chercher une solution approchée d'un problème difficile en un temps polynomial. Ces approches considèrent comme point de départ un ensemble de solutions (population de solutions) initiales. L'algorithme utilise des méthodes de sélection des solutions à travers une fonction d'évaluation qui permet de noter chaque individu selon son adaptation au problème. Les individus sélectionnés participent à la reproduction de nouvelles solutions par un ensemble d'opérations (croisement et mutation). Les étapes de reproduction des populations de solutions se déroulent jusqu'à ce que l'algorithme atteigne un critère d'arrêt par exemple après un certain temps de calcul à ne pas dépasser [56]. Ces étapes sont représentés dans la Figure II.6

Schéma récapitulatif :

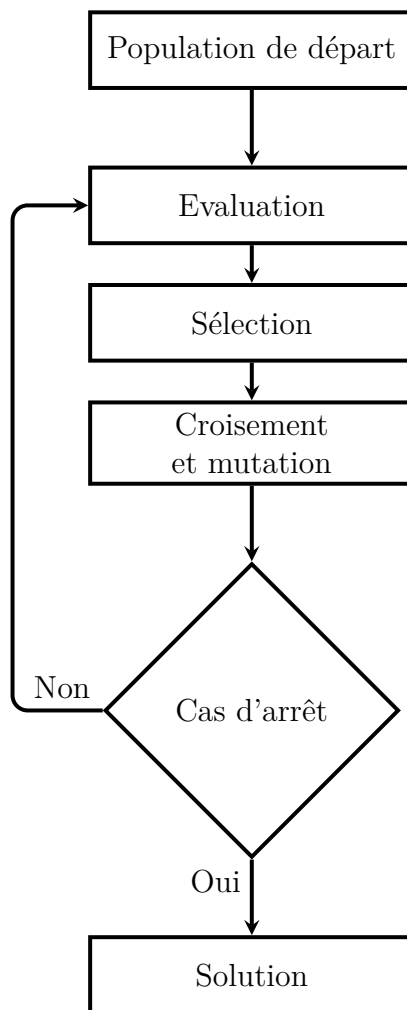


FIGURE II.6 – Algorithme génétique général

— Colonies de Fourmis

L'optimisation des colonies de fourmis est une méta-heuristique basée sur la population pour la résolution de problèmes d'optimisation combinatoire difficiles. Elle s'appelle ainsi en raison de son inspiration : il a été démontré expérimentalement que les fourmis sont capables de trouver le chemin le plus court entre leur nid et une source de nourriture en exploitant collectivement la phéromone qui est une substance chimique qu'elles déposent sur leur chemin. A chaque fois qu'une fourmi se déplace, elle laisse une trace de son passage. Les autres fourmis, qui veulent avoir de la nourriture et qui rencontrent le chemin sur lequel la phéromone est forte, vont arrêter leurs déplacements aléatoires et emprunter ce chemin.

Cette méta-heuristique imite ce comportement tel que les agents artificiels d'ACO

(Ant colony optimization), appelés aussi fourmis artificielles, déposent la phéromone artificielle sur le graphe du problème qu'elles résolvent. La quantité de cette phéromone déposée par chaque fourmi artificielle décide de la qualité de la solution produite. Elle inclut aussi la possibilité d'évaporation des phéromones, c'est-à-dire diriger sa recherche vers de nouvelles directions en oubliant les anciennes décisions [35].

Cet algorithme a été à l'origine utilisé pour la résolution du problème de voyageur de commerce. Ensuite, il a été appliqué à de nombreux problèmes d'Optimisation combinatoire y compris les problèmes d'ordonnancement, le problème d'open shop scheduling et autres. [20].

II.6 Conclusion

Dans ce chapitre, nous avons présenté les problèmes d'optimisation combinatoire et quelques exemples réels de ces problèmes ainsi que leurs méthodes de résolutions. Ces méthodes sont réparties en deux grandes classes : La première classe contient les méthodes exactes qui garantissent l'optimalité de la solution trouvée et la deuxième partie présente les méthodes approchées (heuristique et méta-heuristique) qui offrent de bons résultats rapidement mais ne garantissent pas l'optimalité de ces résultats. Le choix de l'approche suivie pour la résolution de tels problèmes dépend généralement de problème lui-même et sa taille, sa nature, etc.

Le chapitre suivant est consacré à la description de la solution proposée en vue d'un déploiement optimal d'un RCSF dédié à la surveillance. Nous décrivons la formalisation du problème de déploiement sous forme de deux BILP (Binary Integer Linear Program) ainsi que l'implémentation et l'évaluation des performances de ces derniers.

Implémentation et évaluation des performances

III.1 Introduction

Dans ce chapitre, nous proposons un déploiement déterministe multi-objectif de nœuds capteurs et de nœuds relais dans une zone d'intérêt, à savoir un site stratégique (par ex., site pétrolier ou site nucléaire), en utilisant la programmation linéaire. Les nœuds sentinelles, en Anglais Sentinel Nodes (SNs), sont déployés à la frontière du site à surveiller. Leur unique rôle est de générer une alerte en cas de détection d'intrusion. L'acheminement des alertes, jusqu'au sink, est assuré par les nœuds relais, en Anglais Relay Nodes (RNs), qui sont déployés à l'intérieur du site.

Les objectifs sont traités selon la méthode lexicographique II.4.1.1. Le premier objectif est de minimiser le nombre de sauts, appelé k , depuis le nœud sentinelle le plus éloigné jusqu'au Sink, et d'imposer que la longueur de chaque chemin, depuis les autres nœuds sentinelles jusqu'au Sink, soit inférieure ou égale à k . Le second objectif est la minimisation du nombre total de nœuds relais déployés, et ce afin de réduire le coût de déploiement du RCSF.

La résolution successive des deux programmes linéaires, nous permet d'obtenir les positions optimales des SNs et RNs. Ces positions vont être employées pour construire le graphe de connectivité réseau sur lequel on exécute l'algorithme de Floyd [23] pour déterminer le plus court chemin, en termes de nombre de sauts, depuis chaque nœud sentinelle jusqu'au Sink. Ces chemins serviront à router, jusqu'au sink, les alertes générées par les nœuds sentinelles.

Dans ce Chapitre, nous allons décrire et évaluer notre solution au problème de déploiement déterministe d'un RCSF dédié à la surveillance de sites sensibles.

III.2 Problème de placement optimal des nœuds

Considérons une zone d'intérêt (ou *site*) à surveiller à l'aide d'un Réseau de Capteurs Sans Fil. Les nœuds capteurs et les nœuds relais sont déployés d'une manière déterministe afin que chaque point de la frontière soit surveillé par au moins un nœud capteur sentinelle, et à ce qu'il existe un chemin, constitué d'un nombre de sauts limité (minimum de relais), depuis chaque nœud sentinelle jusqu'au sink.

| | | | | | |
|------|------|----|----|----|----|
| | 20 | 21 | 22 | 23 | 24 |
| | 15 | 16 | 17 | 18 | 19 |
| Y(m) | 10 | 11 | 12 | 13 | 14 |
| | 5 | 6 | 7 | 8 | 9 |
| | 0 | 1 | 2 | 3 | 4 |
| | X(m) | | | | |

FIGURE III.1 – Discrétisation sous forme de grille carré de l'espace de déploiement des nœuds.

L'espace de déploiement du site est discrétisé. Le site est modélisé par une grille carré à deux dimensions, à cause de la difficulté d'exploiter toutes les positions possibles dans une zone bien déterminée. Les centres des mailles de la grille ont pour indice i et représentent les emplacements candidats où les nœuds peuvent être déployés.

L'ensemble des mailles matérialisant la frontière de l'espace considéré est noté P . Pour qu'un point $p \in P$ (centre de la maille) soit couvert, il faut que toute la maille soit couverte (les quatre points constituant une maille). Les mailles sont numérotées de (0 à $(n - 1)$) comme le montre la Figure III.1. " n " est le nombre de mailles dans la grille. Dans le cas de la Figure III.1, ($n = 25$). Les coordonnées cartésiennes d'une position i seront calculées comme suite [52] :

$$\left\{ \begin{array}{l} x_i = (i \bmod Grid) pas + \frac{pas}{2} \\ y_i = (i \text{ Div } Grid) pas + \frac{pas}{2} \end{array} \right\}$$

Où,

— $Grid = \sqrt{n}$.

— pas est la distance entre les centres des deux mailles adjacentes.

— mod et div représentent respectivement le reste et le quotient de la division entière.

Les nœuds capteurs S seront déployés au niveau de la frontière de la grille, c-à-d, sur les mailles de bordure, et les nœuds relais R seront déployés à l'intérieur de la grille.

Les nœuds capteurs et les nœuds relais établissent un réseau multi-sauts qui doit être totalement connecté au Sink. Ce dernier est placé aléatoirement à l'intérieur de la grille. Deux nœuds sont directement connectés si la distance entre eux est inférieure ou égale au rayon de communication R_c .

Nous allons opter pour une méthode multi-objectif avec ordonnancement des objectifs selon la méthode lexicographique II.4.1.1. Nous avons proposé un BILP (Binary Integer Linear Program) pour minimiser le nombre de sauts (appelons le k), depuis le nœud sentinelle le plus éloigné jusqu'au Sink, et d'imposer que la longueur de chaque chemin, depuis les autres nœuds sentinelles jusqu'au Sink, soit inférieure ou égale à k . La résolution du premier BILP nous permet d'obtenir un k minimal que nous injectons dans le deuxième BILP. La résolution de ce dernier nous donne le nombre de relais minimal à déployer.

III.3 Programmation linéaire

La programmation linéaire est une technique mathématique d'optimisation (minimisation ou maximisation) dont la fonction objectif et toutes les contraintes sont linéaires. Dans ce cas le problème est noté LP (Linear Program) [10]. Un tel problème s'écrit sous la forme suivante :

$$\begin{array}{ll} \text{Minimize} & f_0(x) \\ \\ \text{subject to} & g_i(x) \leq 0 \\ & i = 1, 2, \dots, m \end{array}$$

- f_0 est la fonction objectif.
- $g_i(i = 1, 2, \dots, m)$ sont les contraintes.
- $x = (x_1, x_2, \dots, x_n)$ est le vecteur qui représente les variables d'optimisation.

La solution de ce problème fourni par $f_0(x)$ est la plus petite valeur possible dont le vecteur x satisfait toutes les contraintes. Si l'objectif ou l'une des contraintes ne sont pas linéaires, le problème s'appelle Programme non linéaire (NLP). Il existe plusieurs méthodes pour la résolution efficace des LP tel que l'algorithme du Simplex

qui permet de trouver l'optimum global de ces *LP*. Cet algorithme est largement utilisé et implémenté dans divers solveurs tel que *CPLEX*.

Un *PLNE* pour programme linéaire en nombres entiers (*ILP* "Integer Linear Program") est un problème linéaire avec une contrainte supplémentaire où tous les variables doivent avoir des valeurs entières. Le problème est appelé Programme linéaire mixte en nombres entiers (*MILP*) si seulement certaines variables prennent des valeurs entières. Si les variables ne peuvent être que 0 ou 1 (Booléennes), on parle alors de Programme linéaire binaire (*BILP*). Ces problèmes sont NP-complet [10] et ne peuvent pas être résolus efficacement en un nombre limité d'itérations tels que les *LP*.

III.4 Le Solveur *CPLEX*

CPLEX prend son nom de la méthode du Simplex implémentée en langage C, bien qu'il prenne actuellement en charge d'autres types d'optimisation mathématique et offre des interfaces autres que le langage C tel que Java, C++, MATLAB, etc. Il a été développé en premier par l'équipe de Robert Bixby pour obtenir un solveur performant utile à la résolution des instances du problème de voyageur de commerce (en Anglais, Travelling Salesman Problem -TSP-) de grande taille. Jusqu'à la version 6.0, il a été commercialisé par la société CPLEX. En 1996, cette société a été rachetée par ILOG qui ensuite a été acquise par IBM en janvier 2009. L'outil *IBM ILOG CPLEX* Optimisation Studio complet comprend l'optimiseur *CPLEX* pour la programmation mathématique, l'optimiseur CP (constrained programming) pour la programmation par contraintes [39], un langage pour la modélisation des problèmes d'optimisation OPL (Optimization Programming Language) et un IDE étroitement intégré.

CPLEX est utilisé pour résoudre plusieurs types de la programmation mathématique :

- Problème linéaire (LP : Linear program).
- Problèmes de programmation quadratique (QP : Quadratic Programming) où on trouve des termes du deuxième degré dans la fonction objectif du programme.
- Problèmes de programmation en nombres entiers (ILP : Integer Linear Program) où les variables doivent prendre des valeurs entières.
- Problèmes de programmation mixte en nombres entiers (MILP : Mixed Integer linear Program) où certaines des variables doivent avoir des valeurs entières.
- Problème de programmation binaire (BILP : Binary Integer linear program) où les variables ne peuvent être que 0 ou 1 (Booléennes)

III.5 Solution proposée

Nous tenons à rappeler que le modèle de surveillance que nous avons adopté est montré dans la Figure III.2

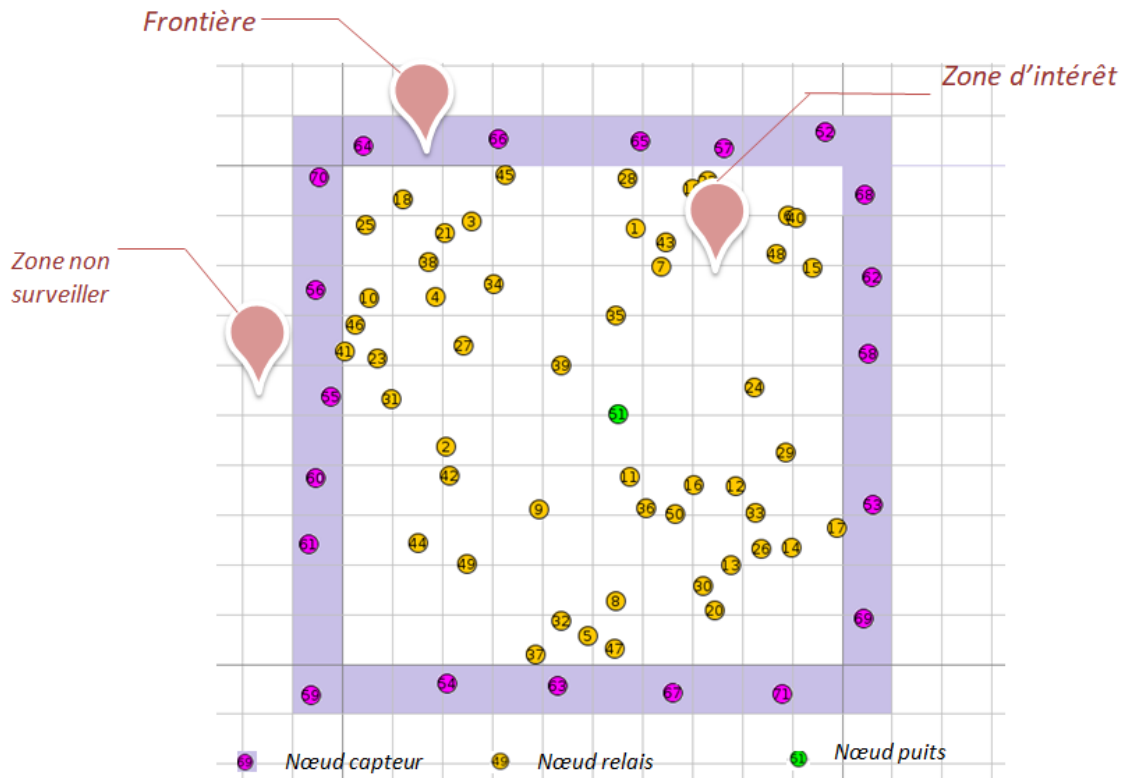


FIGURE III.2 – Modèle de surveillance .

Comme souligné précédemment, notre solution de déploiement se décline en deux BILP.

| Paramètres | Description |
|------------|--|
| V | Ensemble des positions possibles pour les nœuds relais ou sentinelles |
| S | Ensemble des nœuds capteurs(sentinelles) |
| R | Ensemble des nœuds relais |
| P | Ensemble des points à couvrir |
| V'_p | Ensemble des positions candidates appartenant à V pouvant couvrir le point $p \in P$ |
| V_i | Ensemble des positions candidates appartenant à V pouvant communiquer directement avec un nœud capteur ou relais déployé à la position $i \in V$ |
| V^1 | Ensemble des positions candidates appartenant à V pouvant communiquer directement avec le Sink |
| K | Nombre de sauts maximum autorisé |

Tableau III.1 – Les paramètres communs aux deux BILP.

III.5.1 BILP sans contraintes de minimisation du nombre total de nœuds relais

L'objectif de ce programme linéaire est de minimiser le nombre de sauts depuis le nœud sentinelle (SN) le plus éloigné dans le RCSF jusqu'au Sink.

III.5.1.1 Les variables de décision

$$\mathbf{x}_i^k = \begin{cases} 1 & \text{Si un nœud capteur est déployé à la position } i, \text{ à } k \text{ sauts du Sink} \\ 0 & \text{Sinon} \end{cases}$$

$$\mathbf{r}_i^k = \begin{cases} 1 & \text{Si un nœud relais est déployé à la position } i, \text{ à } k \text{ sauts du Sink} \\ 0 & \text{Sinon} \end{cases}$$

$$\mathbf{y}_k = \begin{cases} 1 & \text{Si il existe un nœud relais déployé à } k \text{ sauts du Sink} \\ 0 & \text{Sinon} \end{cases}$$

$$\mathbf{z}_i = \begin{cases} 1 & \text{Si un nœud capteur est déployé à la position } i \\ 0 & \text{Sinon} \end{cases}$$

$$t_i = \begin{cases} 1 & \text{Si un nœud relais est déployé à la position } i \\ 0 & \text{Sinon} \end{cases}$$

III.5.1.2 Fonction objectif

$$\min \sum_{k=1}^K y_k$$

III.5.1.3 Les contraintes

$$\sum_{k=1}^K x_i^k + r_i^k \leq 1 \quad i \in V \quad (\text{III.1})$$

$$\sum_{k=1}^K \sum_{i \in V'_p} x_i^k \geq 1 \quad p \in P \quad (\text{III.2})$$

$$x_i^k \leq \sum_{j \in V_i} r_j^{k-1} \quad i \in V \setminus V^1, \quad k = 2 \dots K \quad (\text{III.3})$$

$$r_i^k \leq \sum_{j \in V_i} r_j^{k-1} \quad i \in V \setminus V^1, \quad k = 2 \dots K \quad (\text{III.4})$$

$$x_i^1 = r_i^1 = 0 \quad i \in V \setminus V^1 \quad (\text{III.5})$$

$$y_k \geq r_i^k \quad i \in V, \quad k = 1 \dots K \quad (\text{III.6})$$

$$y_k \leq \sum_{i \in V} r_i^k \quad k = 1 \dots K \quad (\text{III.7})$$

$$y_k \leq y_{k-1} \quad k = 2 \dots K \quad (\text{III.8})$$

$$z_i \geq x_i^k \quad i \in V, \quad k = 1 \dots K \quad (\text{III.9})$$

$$z_i \leq \sum_{k=1}^K x_i^k \quad i \in V \quad (\text{III.10})$$

$$t_i \geq r_i^k \quad i \in V, \quad k = 1 \dots K \quad (\text{III.11})$$

$$t_i \leq \sum_{k=1}^K r_i^k \quad i \in V \quad (\text{III.12})$$

$$\sum_{i \in V} z_i \leq |S| \quad (\text{III.13})$$

III.5.1.4 Description des contraintes

— La contrainte (1) exige que chaque maille doit être occupée par au plus un nœud de type relais ou capteur situé à k sauts du Sink.

- La contrainte (2) assure que chaque point est couvert par au moins un nœud capteur qui est situé à k sauts du Sink.
- La contrainte (3) garantit que chaque nœud sentinelle, non voisin du Sink et situé à k sauts du Sink, a au moins un nœud voisin de type relais situé à $K - 1$ sauts du Sink.
- La contrainte (4) garantit que chaque nœud relais, non voisin du Sink situé à k sauts du Sink, a au moins un nœud voisin de type relais situé à $K - 1$ sauts du Sink.
- La contrainte (5) évite l'incohérence du modèle, c-a-d, aucun nœud de type sentinelle ou type relais n'est voisin du Sink que s'il est situé à 1 saut du Sink.
- La contrainte (6) détermine la longueur des chemins des nœuds relais vers le Sink, s'il y a un nœud relais dans la i^{eme} position situé à k sauts du Sink alors il existe un chemin de longueur k depuis un nœud relais vers le Sink.
- La contrainte (7) assure que s'il y a un chemin de longueur k sauts cela signifie qu'il y a au moins un nœud relais dans la i^{eme} position situé à k sauts du Sink et cela pour éviter l'incohérence du modèle.
- La contrainte (8) garantie que l'existence d'un chemin de longueur k sauts induit la présence d'un chemin de longueur $k - 1$ sauts.
- La contrainte (9) assure que s'il existe un nœud sentinelle dans la i^{eme} position situé à k sauts du Sink, il sera réellement déployé dans la i^{eme} position.
- La contrainte (10) assure que si un nœud sentinelle est déployé à la i^{eme} position alors il y a forcément un nœud sentinelle à la i^{eme} position situé à k sauts du Sink pour éviter l'incohérence du modèle.
- La contrainte (11) assure que s'il y a un nœud relais dans la i^{eme} position situé à k sauts du Sink, alors il sera réellement déployé dans la i^{eme} position.
- La contrainte (12) assure que si un nœud relais est déployé à la i^{eme} position, alors il y a forcément un nœud relais à la i^{eme} position situé à k sauts du Sink pour éviter l'incohérence du modèle.
- La contrainte (13) exige que le nombre de nœuds capteurs déployés soit inférieur au nombre des nœuds capteurs disponibles.

III.5.1.5 Evaluation des performances du BILP

| Paramètre | Valeur |
|-----------------------------|--------------|
| Surface du site | 40 000 m^2 |
| Taille d'une maille | 20 m |
| Nombre de nœuds capteurs | 36 |
| Rayon de capture | 15 m |
| Rayon de communication | 30 m |
| Nombre de sauts maximal (K) | 9 |

Tableau III.2 – Paramètres d'exécution du BILP.

La Figure III.3 montre le résultat du déploiement des nœuds capteurs, nœuds relais et le Sink après la résolution du BILP, en utilisant le solveur *CPLEX*.

Dans cet exemple, nous avons utilisé une grille de 10×10 :

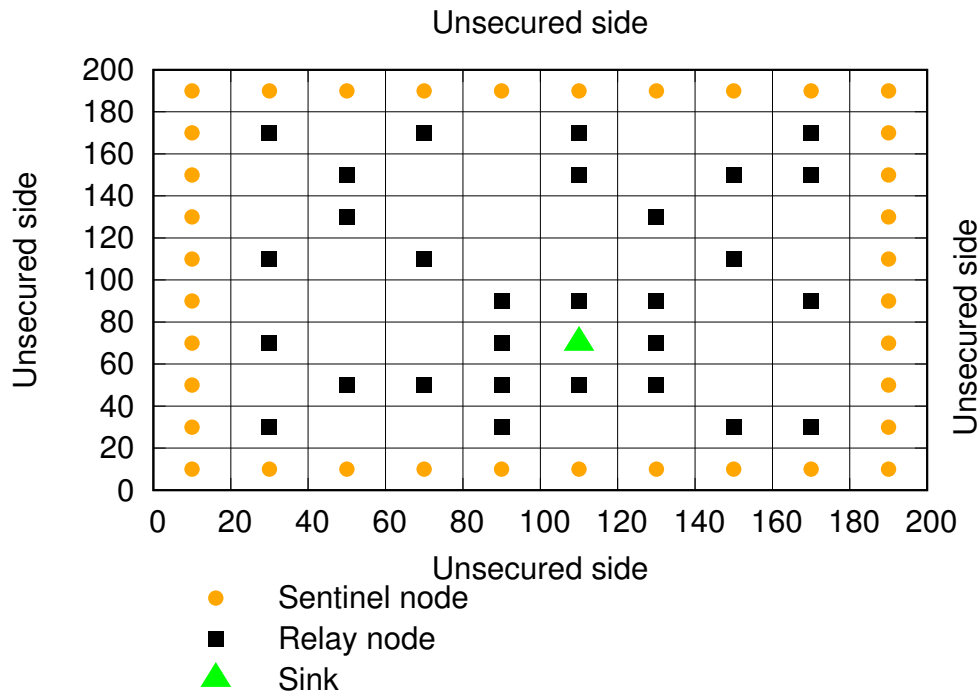


FIGURE III.3 – Déploiement des nœuds après résolution du BILP.

La Figure III.3 montre bien que toutes les mailles de la bordure sont couvertes. Nous obtenons donc une couverture totale de la frontière ainsi qu'un chemin vers le Sink, depuis chaque nœud sentinelle (étant donné que nous avons considéré un R_C égal à la distance entre les centres de deux mailles adjacentes).

| Nombre de sauts | Nombre de chemins |
|----------------------------------|-------------------|
| 3 | 6 |
| 4 | 6 |
| 5 | 9 |
| 6 | 15 |
| Nombre total des nœuds relais 29 | |
| Nombre de sauts maximal 6 | |
| Nombre moyenne de sauts 4,92 | |

Tableau III.3 – Statistiques obtenus après la résolution du BILP 1

Le Tableau III.3, montre quelques statistiques obtenues après la résolution du BILP. On peut constater que le nombre de relais optimal utilisé est égal à 29 relais, et que plus de la moitié des chemins varient entre 5 et 6 sauts. Le nombre moyen de sauts vaut 4,92.

III.5.2 BILP avec les contraintes de minimisation du nombre total de nœuds relais

L'objectif de ce modèle est de minimiser le nombre de nœuds relais total déployés afin d'impacter le coût de déploiement du RCSF et éventuellement réduire encore la longueur moyenne des chemins entre les nœuds sentinelles et le nœud puits (Sink).

Nous tenons à rappeler que ce BILP utilise la valeur du nombre de sauts du plus long chemin entre un nœud sentinelle et le nœud puits, obtenu après la résolution du précédent BILP (III.5.1)

III.5.2.1 Les variables de décision

Nous notons que ce BILP utilise seulement quatre variables de décision parmi les cinq utilisées par le premier BILP, à savoir x_i^k, r_i^k, z_i, t_i (III.5.1).

III.5.2.2 Fonction objectif

$$\min \sum_{i \in V} t_i$$

III.5.2.3 Les contraintes

Ce BILP utilise les mêmes contraintes que le premier BILP, mis à part (6,7,8) (III.5.1.4)

III.5.2.4 Description des contraintes

Il est à noter que la description des contraintes utilisées dans ce BILP est la même que celle de la Section III.5.1.4

III.5.2.5 Evaluation des performances BILP

| Paramètre | Valeur |
|--|--------------|
| Surface du site | 40 000 m^2 |
| Taille d'une maille | 20 m |
| Nombre de nœuds capteurs | 36 |
| Rayon de capture | 15 m |
| Rayon de communication | 30 m |
| Nombre de sauts maximal obtenus en résolvant le BILP 1 | 6 |

Tableau III.4 – Paramètres d'exécution du BILP.

La Figure III.4, montre le résultat du déploiement des nœuds capteurs, nœuds relais et Sink après la résolution du BILP, et ce en utilisant le solveur *CPLEX*. Comme pour le BILP précédent, nous avons utilisé une grille de 10×10 .

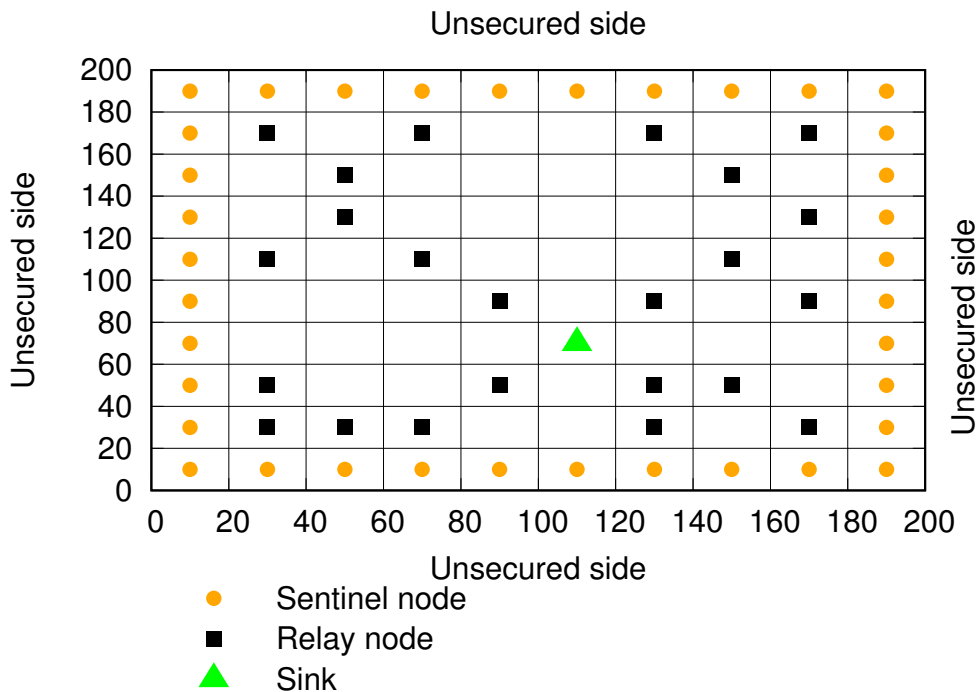


FIGURE III.4 – Déploiement des nœuds après résolution du deuxième BILP

| Nombre de sauts | Nombre de chemins |
|----------------------------------|-------------------|
| 3 | 6 |
| 4 | 10 |
| 5 | 7 |
| 6 | 13 |
| Nombre total des nœuds relais 23 | |
| Nombre moyenne de sauts 4,75 | |

Tableau III.5 – Statistiques obtenus après la résolution du BILP 2

Le Tableau III.5, montre quelques statistiques obtenues après la résolution du BILP. On peut constater que le nombre de relais optimal est égal à 23 relais (6 relais de moins par rapport à la première étape) et que plus de la moitié des chemins varient entre 5 et 6 sauts. Par contre, le nombre moyen de sauts vaut 4,75 (on constate une légère réduction par rapport à la première étape).

III.5.3 Temps d'exécution des BILPs

Le problème de placement optimal des nœuds est un problème qui a été prouvé NP-difficile pour la plupart des formulations de déploiement des nœuds. Les résultats obtenus après plusieurs expériences confirment la complexité de ce problème. Nous remarquons à travers le Tableau III.6 que le temps de la résolution des deux BILPs augmente exponentiellement avec l'augmentation de la taille de la grille utilisée.

| Taille de la grille | BILP 1 Temps d'exécution | BILP 2 Temps d'exécution |
|---------------------|--------------------------|--------------------------|
| 11 x 11 | 00 :00 :18 | 00 :02 :22 |
| 12 x 12 | 00 :00 :49 | 00 :05 :38 |
| 13 x 13 | 00 :00 :59 | 09 :58 :56 |
| 14 x 14 | 00 :00 :93 | 39 :35 :98 |
| 15 x 15 | 00 :01 :04 | ≥ 20 heure |

Tableau III.6 – Temps d'exécution des BILPs

III.6 Validation des résultats par simulation

Nous évaluons notre approche au moyen de simulations, en utilisant le simulateur *CASTALIA*, qui repose sur la Plateforme *OMNeT++* [46]. Nous utilisons trois métriques, à savoir le nombre de sauts moyen, la latence moyenne et la consommation totale d'énergie dans le RCSF, pour comparer les performances de notre méthode de déploiement à celle d'une approche que nous appellerons approche semi-aléatoire. Celle-ci consiste à déployer les nœuds relais de manière aléatoire, alors que les nœuds sentinelles sont déployés d'une manière déterministe. Cette approche assure, comme

notre approche déterministe, l'existence d'au moins un chemin jusqu'au sink, depuis tout nœud sentinelle (voir la Figure III.5). La comparaison est faite en variant le nombre de sources d'alerte (nombre de sentinelles ayant détectées une intrusion). Nous notons que les résultats discutés ci-dessous sont la moyenne de 100 simulations.

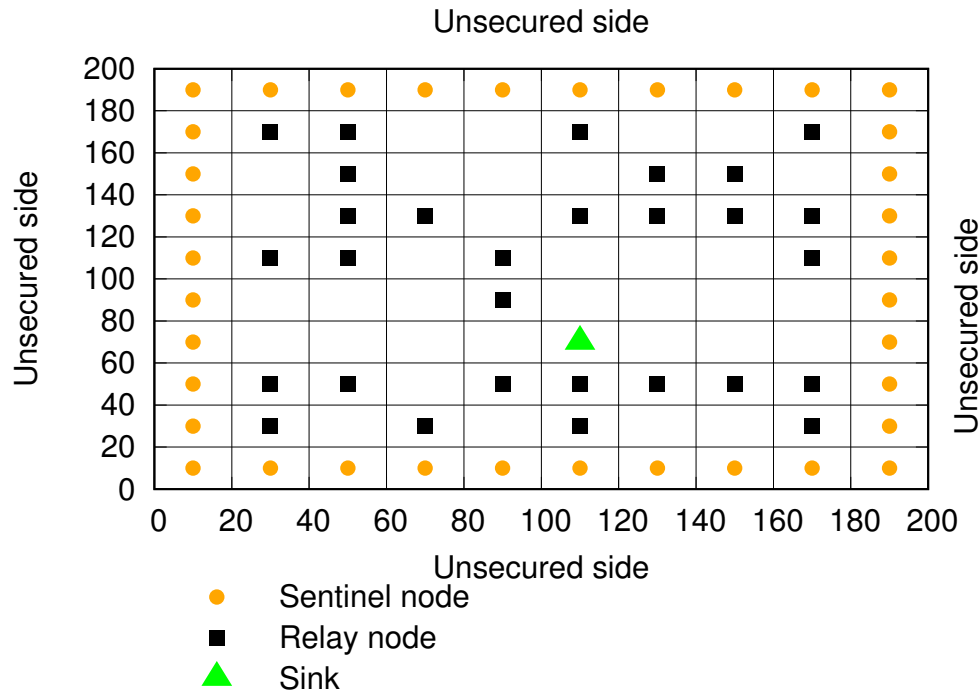


FIGURE III.5 – Déploiement semi-aléatoire

III.6.1 Evaluation du nombre de sauts moyen

La Figure III.6 montre le nombre de sauts moyen en fonction du nombre de sources d'alerte.

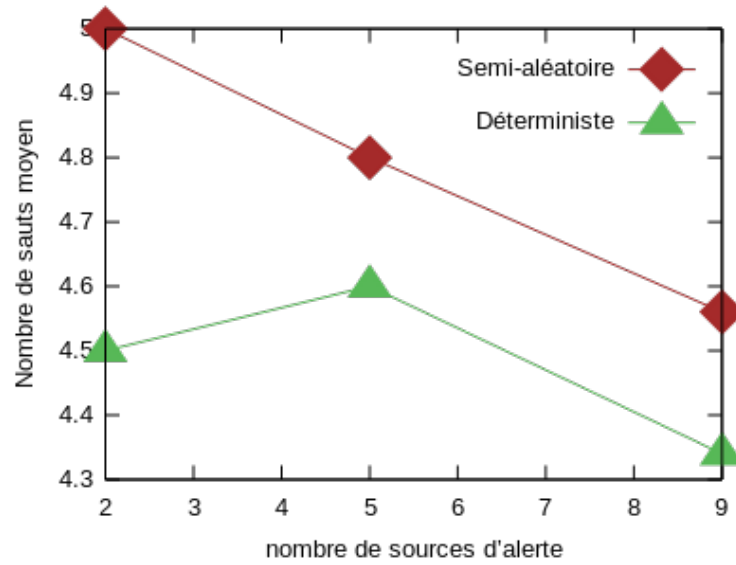


FIGURE III.6 – Nombre de sauts moyen parcourus par une alerte en fonction du nombre de sources d’alerte

Comme le montre la Figure III.6, l’approche déterministe a toujours un nombre de sauts moyen inférieur à celui de la méthode semi-aléatoire. Ce résultat confirme que la méthode déterministe offre des chemins plus courts que ceux de la méthode semi-aléatoire.

III.6.2 Évaluation de la latence dans le réseau

La Figure III.7 montre la latence moyenne des paquets (en *ms*) en fonction du nombre de sources d’alerte.

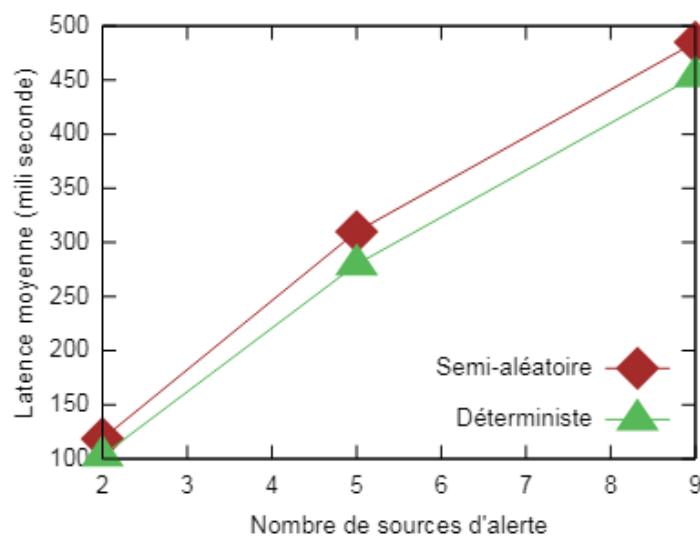


FIGURE III.7 – Latence moyenne en fonction du nombre de sources d’alertes

Les résultats obtenus montrent que la latence augmente quand le nombre de sources d'alerte augmente. Une faible latence garantit que l'alerte générée à la suite de la détection d'une intrusion, est transmise à la station de base dans un délais raisonable. Ceci assure une meilleure qualité de surveillance.

III.6.3 Évaluation de la consommation d'énergie dans le réseau

Enfin, la Figure III.8 montre la consommation d'énergie totale (en miliJoules (mJ)) en fonction du nombre de sources d'alerte.

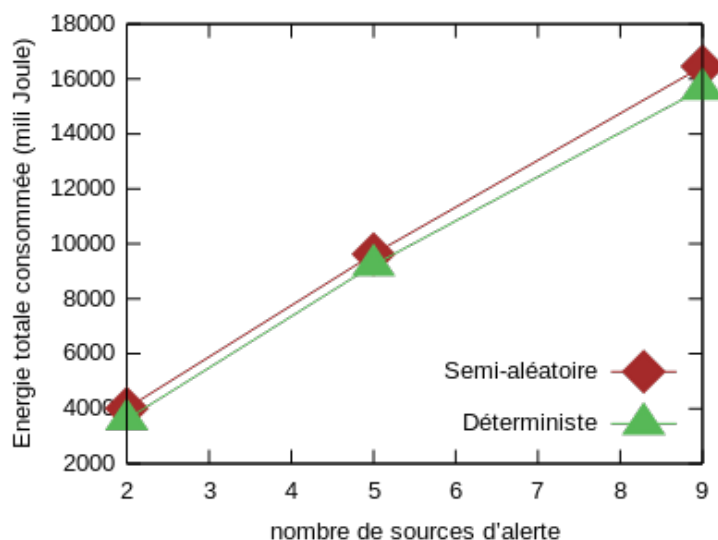


FIGURE III.8 – Énergie totale consommée dans le réseau en fonction du nombre de sources d'alerte.

Comme illustré dans la Figure III.8, les performances de notre solution en termes de consommation énergétique sont meilleures que celles de la méthode semi-aléatoire. Les performances de notre méthode peuvent être expliquées par le fait que nous avons un nombre de sauts limité. La réduction de la consommation énergétique a pour conséquence l'amélioration de la durée de vie des capteurs, et par conséquent la durée de vie du réseau. Ainsi, avec une méthode déterministe, la durée de vie du réseau peut être nettement améliorée relativement à un déploiement semi-aléatoire. En outre, la couverture complète de la zone de surveillance et la connectivité optimale du réseau sont également garanties dans notre méthode à tout moment de la durée de vie du RCSF.

III.7 Conclusion

Dans ce chapitre nous avons proposé une stratégie de déploiement déterministe des nœuds d'un RCSF, qui assure la couverture totale de la frontière et la connectivité réseau avec un nombre de sauts limité, tout en minimisant le coût de déploiement en termes de nœuds relais déployés.

Nous avons formulé notre problème de déploiement sous forme de deux modèles BILP que nous avons résolus en utilisant le solveur *CPLEX* qui fournit une solution exacte.

Nos expérimentations ont montré que notre approche donne un bon déploiement avec un coût minimal.

Les résultats de la simulation montrent également l'impact important de la minimisation du nombre de sauts sur la latence et la consommation d'énergie dans le RCSF.

Conclusion générale et perspectives

Dans ce mémoire, nous avons abordé une problématique très importante au sein des Réseaux de Capteurs Sans Fil, à savoir le déploiement des nœuds formant un RCSF. En effet, il a été démontré que le déploiement des nœuds impacte les performances d'un RCSF, notamment en termes de couverture, connectivité, énergie et latence.

Nous tenons à rappeler que le problème du déploiement des nœuds d'un RCSF fait partie des problèmes d'optimisation difficile et sa modélisation mathématique est une tâche ardue. Le modèle réalisé doit être le plus simple possible et doit refléter la difficulté du problème sans le compliquer davantage.

En ce qui nous concerne, nous avons considéré le problème du déploiement des nœuds comme étant un problème d'optimisation multi-objectif et nous l'avons formalisé sous forme de deux BILP (Binary Integer Linear Program). Ces modèles sont rangés par ordre selon la méthode lexicographique où chaque modèle est résolu en utilisant le solveur *CPLEX* pour avoir une solution exacte de chaque modèle.

Notre but est de trouver les positions optimales des nœuds Sentinelles (en Anglais, Sentinel Nodes -SNs-) et des nœuds relais (en Anglais, Relay Nodes -RNs). Les SNs sont déployés sur la frontière du site à surveiller afin d'assurer la couverture de chaque point de la frontière et de détecter par conséquent toute intrusion. Les RNs sont déployés à l'intérieur du site et servent à relayer, jusqu'au sink, les alertes générées par les SNs.

La fonction objectif du premier BILP consiste à minimiser le nombre de sauts k du chemin le plus long, depuis un nœud sentinelle jusqu'au nœud puits (sink), et ensuite imposer que la longueur des autres chemins, depuis les autres nœuds sentinelles jusqu'au sink, soit inférieure à k . Le but de la limitation du nombre de sauts au sein du RCSF est d'impacter positivement la consommation de l'énergie et la latence au sein du réseau, lors du routage des alertes générées par les nœuds sentinelles jusqu'au sink.

La fonction objectif du dixième BILP consiste à minimiser le nombre total de nœuds relais déployé afin de réduire le coût du déploiement, et ce en tenant compte du nombre de sauts k obtenu lors de la résolution du premier BILP.

Les positions optimales obtenues après la résolution des deux BILP vont servir à construire le graphe de connectivité réseau sur lequel on exécute l'algorithme de Floyd [23] pour déterminer le plus court chemin, en termes de nombre de sauts, depuis chaque nœud sentinelle jusqu'au Sink. Ces chemins vont être empruntés par les alertes lors du processus de routage de celles-ci, depuis les nœuds sentinelles jusqu'au sink.

Notre solution de déploiement déterministe des nœuds d'un RCSF dédié à la surveillance, a été évaluée en la comparant à une méthode semi-aléatoire (c.-à-d., les nœuds sentinelles sont déployés d'une manière déterminite alors que les nœuds relais sont déployés d'une manière aléatoire), et ce en utilisant le simulateur *CASTALIA*. Les résultats obtenus en termes de latence et d'énergie sont satisfaisants.

Comme perspectives, nous envisageons de :

- Passer d'une approche exacte à une approche heuristique pour pouvoir traiter des instances de grande taille en un temps raisonnable.
- Prévoir plusieurs chemins disjoints d'un nœud capteur vers le sink (augmenter la résilience du système de surveillance).

Annexes

Architecture de l'application

Notre application permet d'optimiser le déploiement d'un Réseau de Capteurs Sans Fil en vue de limiter la longueur des chemins ainsi que le nombre des nœuds relais déployés, et ce selon la taille de la grille (nombre de mailles constituant la grille) choisi par l'utilisateur. Un nombre de sauts maximal appelé K doit être défini par l'utilisateur.

L'architecture de notre application est représentée dans la Figure 9.

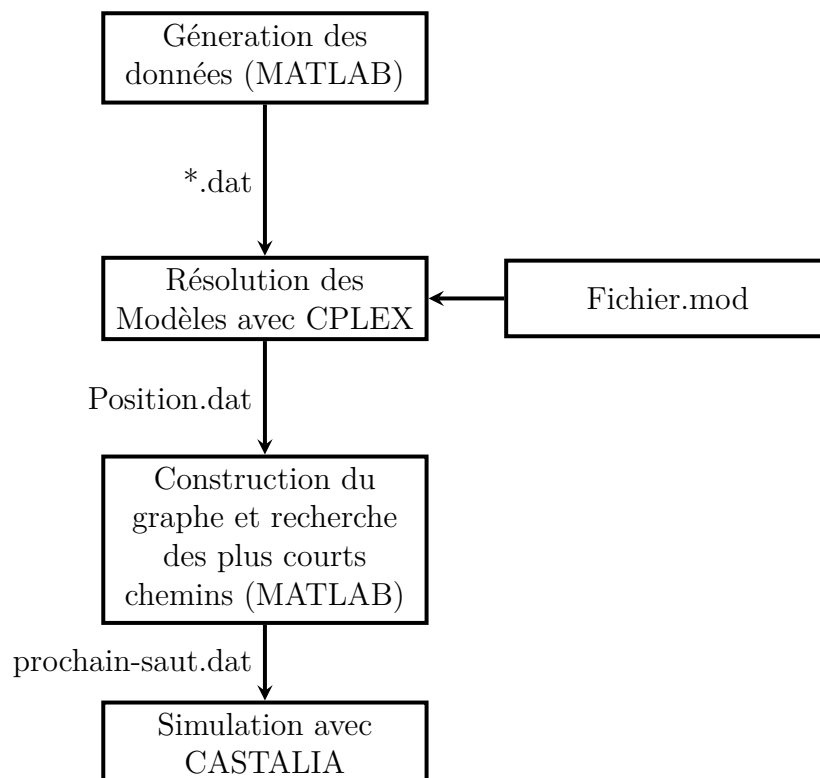


FIGURE 9 – Architecture de l'application

IBM ILOG CPLEX Optimization Studio

1 Présentation

IBM ILOG CPLEX Optimization Studio est le moyen le plus rapide de créer des modèles d'optimisation efficaces et des applications de pointe permettant de traiter de multiples problèmes. Il regroupe un ensemble d'outils pour la programmation mathématique et la programmation par contraintes. Il associe :

- Un environnement de développement intégré (Integrated Development Environment IDE) nommé Cplex Studio IDE (sous Windows) ou oplide (sous Linux).
- Un langage de modélisation : le langage OPL (Optimization Programming Language).
- deux solveurs : IBM ILOG CPLEX pour la programmation mathématique (résolution de programmes linéaires en nombres fractionnaires, mixtes ou entiers et de programmes quadratiques) et IBM ILOG CP Optimizer pour la programmation par contraintes.

Par défaut c'est le solveur CPLEX qui est activé.

Les composants de la suite d'optimisation ILOG sont illustrés dans la Figure 10

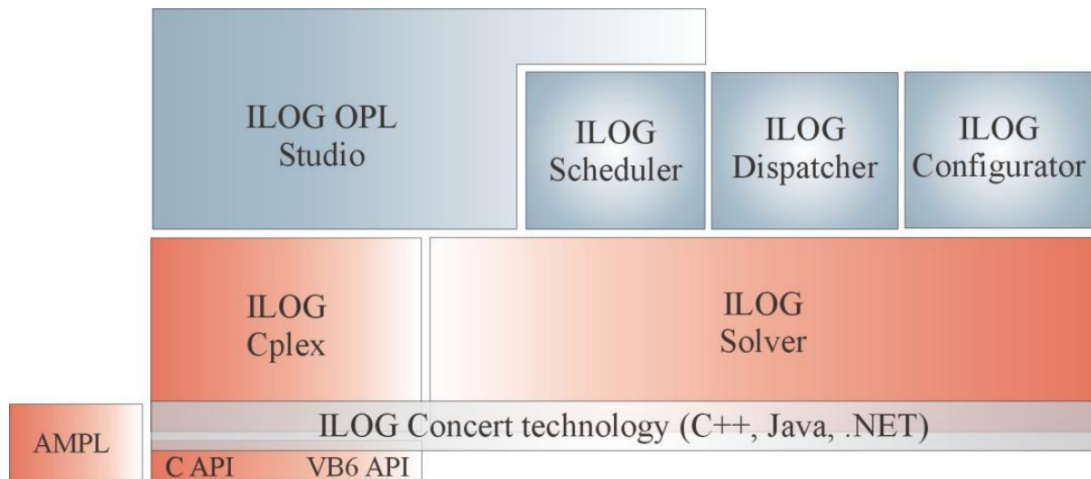


FIGURE 10 – Les composants de la suite d’optimisation ILOG

- **ILOG CPLEX** : Le cœur du système résout des problèmes de programmation mathématique.
- **ILOG Solver** : La partie principale du système, résout des applications en utilisant la programmation par contraintes.
- **ILOG concert technology** : Les bibliothèques contiennent les fonctionnalités du système. Elles sont disponibles pour les langages C++, Java et .NET.
- **ILOG Scheduler** : Fournit des extensions pour résoudre des problèmes de planification.
- **ILOG Dispatcher** : Fournit des extensions pour la résolution de problèmes de tournées de véhicules.
- **ILOG Configurator** : Ce module contient des utilitaires pour l’optimisation des ventes en ligne (problèmes de E-Commerce).
- **ILOG OPL Studio** : OPL est un langage pour la modélisation des problèmes d’optimisation. Il interagit directement avec les modules ILOG Cplex, ILOG Solver et ILOG Dispatcher.
- **AMPL** : C’est un autre langage pour la modélisation, qui interagit avec le module ILOG CPLEX (AMPL a été développé par les laboratoires Bell).
- **C et VB6 APIs** : Ce sont des bibliothèques pour des utilisateurs du langage C et de l’environnement VB6. Elles permettent de les interfacier avec le module ILOG CPLEX.

1.1 Modes d’utilisation

Il existe deux manières d’utiliser CPLEX. La première consiste à travailler de manière interactive en invoquant un interpréteur de commande dédié. La seconde

consiste à utiliser une API, permettant d'interfacer CPLEX à l'aide d'un langage de programmation(C, du C++ ou du Java).

2 Installation sous Windows

Sous Windows, l'emplacement d'installation par défaut d'IBM ILOG CPLEX Optimization Studio est :

C :\Program Files\IBM\ILOG\CPLEX\Studio[édition]1261 .

Pour une installation Windows 32 bits sur une plateforme Windows 64 bits, l'emplacement d'installation par défaut est :

C :\Program Files (x86)\IBM\ILOG\CPLEX\Studio[édition]1261

où [édition] peut être :

* Preview Edition.

* vide, pour l'édition commerciale standard.

Lors de l'installation, vous avez peut-être sélectionné un emplacement différent ([rép_install]).

IDE

Vous pouvez lancer l'IDE en utilisant une entrée du menu Démarrer :

Démarrer > Tous les programmes > IBM ILOG > CPLEX Optimization Studio [édition] 12.6.1 > CPLEX Studio IDE

où [édition] peut être :

* Preview Edition

* vide, pour l'édition commerciale standard

Informations complémentaires

Pour plus d'informations sur la fonctionnalité CPLEX sur la plateforme Windows, voir [rép_install]\cplex \readmeWindows.html.

Pour plus d'informations sur la fonctionnalité CP Optimizer sur la plateforme Windows, voir [rep_install]\cpoptimizer \readmeWindows.html.

3 Installation sous UNIX, Linux ou Macintosh

Le programme d'installation UNIX / Linux / Macintosh est un fichier exécutable. Pour lancer une installation, vous devez donner à ce fichier un droit d'exécution et vous devez l'exécuter depuis une ligne de commande. Par exemple, lancez la commande ./<nominstall>.bin où <nominstall> est le nom du programme d'installation. Vous pouvez donner l'autorisation d'exécution à un fichier avec la commande `chmod u+x <nom_installation>.bin`.

Sous UNIX, l'emplacement d'installation par défaut d'IBM ILOG CPLEX Optimization Studio est :

/opt/IBM/ILOG/CPLEX_Studio[édition]1261 Sous Linux, l'emplacement d'installation par défaut d'IBM ILOG CPLEX Optimization Studio est :

/opt/ibm/ILOG/CPLEX_Studio[édition]1261

où [édition] peut être :

* Preview Edition

* vide, pour l'édition commerciale standard

Lors de l'installation, vous avez peut-être sélectionné un emplacement différent ([rép_install]).

IDE

l'IDE peut être lancé depuis un terminal Linux.

1. Ouvrez une fenêtre terminal
2. Passez dans le répertoire [rép_install]/opl/oplide
3. Entrez la commande ./oplide

Vous pouvez également entrer le chemin d'accès absolu à partir de n'importe quel emplacement du terminal :

[rép_install]/opl/oplide/oplide

Informations complémentaires Pour plus d'informations sur la fonctionnalité CPLEX sur les plateformes UNIX/Linux, voir [rép_install]/cplex/readmeUNIX.html.

Pour plus d'informations sur la fonctionnalité CP Optimizer sur les plateformes UNIX/Linux, voir [rép_install]/coptimizer/readmeUNIX.html.

4 Le langage OPL (Optimization Programming Language)

Le langage utilisé dans Cplex Studio IDE est OPL. Il s'agit d'un langage de modélisation qui permet d'écrire facilement des programmes linéaires (ou quadratiques) grâce à une syntaxe proche de la formulation mathématique. Par ailleurs, OPL offre à l'utilisateur la possibilité de séparer le modèle des données. De ce fait, un même modèle peut être facilement testé avec différents jeux de données.

OPL fonctionne par projets : pour résoudre un modèle l'utilisateur doit créer un projet OPL dans Cplex Studio IDE qui doit contenir au minimum un fichier "modèle" et un fichier de "configuration d'exécution". En effet chaque projet est constitué de plusieurs types de fichiers :

- Un fichier modèle (.mod) qui contient le modèle à résoudre.

- Un fichier de données (.dat) (facultatif) qui contient les données pour un modèle,
- Un fichier de paramètres (.ops) (facultatif) qui permet de paramétrer le solveur CPLEX,
- Un fichier de configuration de l'exécution (.oplproject) qui indique à l'IDE ce qu'il doit faire quand l'utilisateur demande l'exécution du projet, c'est à dire quel est le modèle à résoudre et quels sont les paramètres et les données (s'il y en a).

4.1 Exemple : création et résolution d'un modèle à 2 variables et 2 contraintes

4.1.1 Notions de base de la syntaxe OPL

OPL reconnaît les types entiers (int), entiers positifs (int+), flottants (float) et flottants positifs (float+).

- Les variables de décision se définissent en utilisant le mot clé dvar suivi de leur type.
- La fonction objectif est précédée du mot clés “minimize” ou “maximize” en fonction du sens d'optimisation,
- Les contraintes sont dans un bloc entre accolades et précédées des mots clés “subject to”.
- les principaux opérateurs numériques et logiques sont :
 - +, -, *, / pour l'addition, la soustraction, la multiplication et la division.
 - div et mod pour la division entière et le modulo.
 - <=, >= et == pour les comparaisons.
- les commentaires sont soit entourés de /* ... */ (comme en C), soit précédé de // (comme en C++) s'ils sont sur une seule ligne.

Remarque : la division réelle (/) n'est autorisée que sur des types float, pour les entiers il faut utiliser div.

4.1.2 Création du projet et du modèle

Tout d'abord, il faut créer un projet OPL dans lequel on pourra définir notre modèle. Pour cela dans Cplex Studio il faut cliquer sur Fichier / Nouveau / Projet OPL. Une fenêtre s'ouvre : entrez un nom de projet, choisissez son emplacement (dossier parent) et cochez “Création d'un modèle” ainsi que “ajouter une configuration d'exécution par défaut”.

Dans le fichier modèle (.mod) on entre le modèle présenté dans la Figure 11

| | |
|--|--|
| Minimiser $3x + 2y$ Sous $x - y \geq 5$ $3x + 2y \geq 10$ | <pre>dvar float x; dvar float y; minimize 3*x + 2*y; subject to { x - y >= 5; 3*x + 2*y >= 10; }</pre> |
| <i>formulation mathématique</i> | <i>formulation dans le langage OPL</i> |

FIGURE 11 – Exemple d’un fichier modèle.

4.1.3 Résolution

Pour lancer la résolution, il faut faire un clic droit sur “Configuration d’exécution” dans l’onglet Projets OPL situé à gauche de la fenêtre principale puis “exécuter / configuration d’exécution par défaut”. Le bouton exécuter dans la barre d’outils permet de lancer une nouvelle fois la dernière configuration exécutée.

Une fois le modèle résolu, plusieurs informations s’affichent dans les onglets situés en bas de la fenêtre principale (sous le fichier modèle). Par exemple (Voire Figure 12) :

- L’onglet “solution” donne des informations sur la solution (optimalité, coût de la fonction objectif, valeur des variables à l’optimalité).
- L’onglet “journal du moteur” affiche la sortie de CPLEX,
- L’onglet “statistique” montre différentes mesures liées à la résolution (nombre d’itérations du simplexe, nombres de nœuds de branchement...).
- Tout en bas à droite de la fenêtre s’affiche le chronomètre qui mesure le temps mis par CPLEX pour résoudre le modèle.

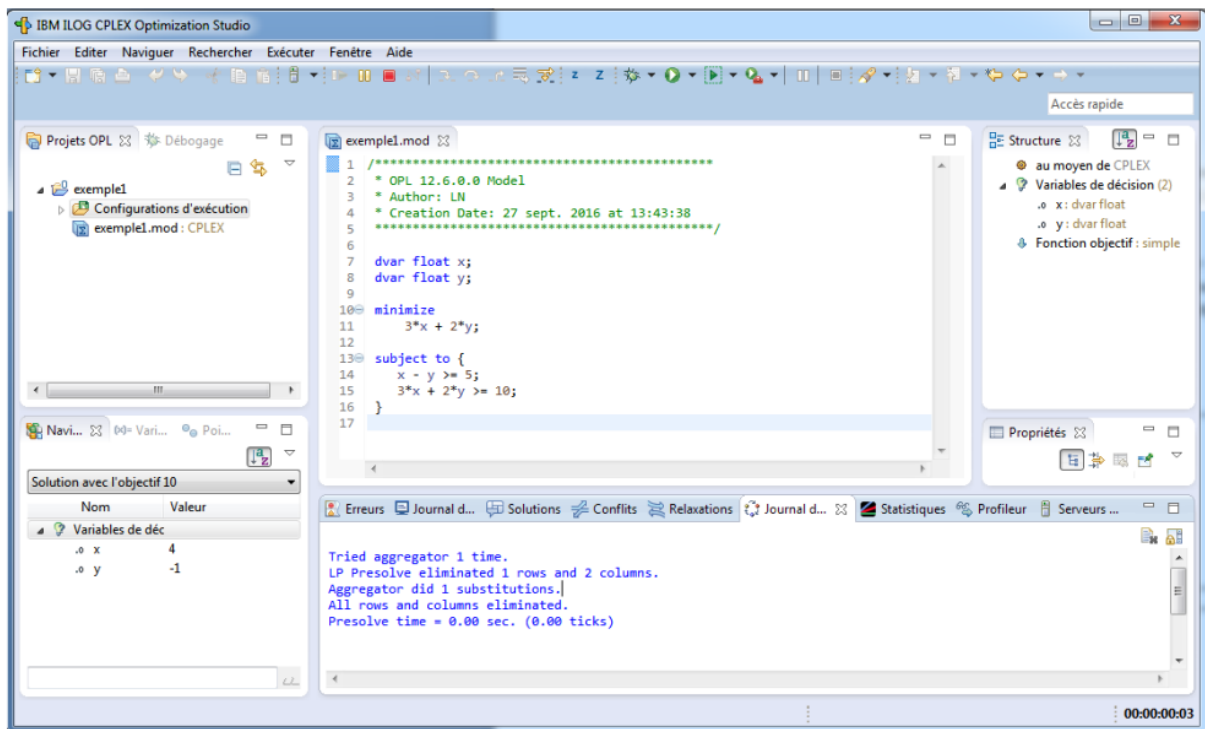


FIGURE 12 – Résolution d'un modèle dans Cplex Studio IDE.

MATLAB (MATrix LABoratory)

1 Présentation

MATLAB est écrit à l'origine en Fortran par C. Moler. Il était destiné à faciliter l'accès au logiciel matriciel développé dans les projets LINPACK et EISPACK. La version actuelle, écrite en C par the MathWorks Inc., existe en version professionnelle et en version étudiant.

MATLAB est un environnement complet, ouvert et extensible pour le calcul et la visualisation. Il dispose de plusieurs centaines (voire milliers, selon les versions et les modules optionnels autour du noyau MATLAB, de fonctions mathématiques, scientifiques et techniques. L'approche matricielle de MATLAB permet de traiter les données sans aucune limitation de taille et de réaliser des calculs numériques et symboliques de façon fiable et rapide. Grâce aux fonctions graphiques de MATLAB, il devient très facile de modifier interactivement les différents paramètres des graphiques pour les adapter selon nos souhaits.

L'approche ouverte de MATLAB permet de construire un outil sur mesure. On peut inspecter le code source et les algorithmes des bibliothèques de fonctions (Toolboxes), modifier des fonctions existantes et ajouter d'autres.

MATLAB comprend aussi un ensemble d'outils spécifiques à des domaines, appelés Toolboxes (ou Boîtes à Outils). Indispensables à la plupart des utilisateurs, les Boîtes à Outils sont des collections de fonctions qui étendent l'environnement MATLAB pour résoudre des catégories spécifiques de problèmes. Les domaines couverts sont très variés et comprennent notamment le traitement du signal, l'automatique, l'identification de systèmes, les réseaux de neurones, les statistiques, l'optimisation, etc.

1.1 Interface de MATLAB

L'interface de MATLAB se compose de plusieurs zones :



FIGURE 13 – L'interface de MATLAB

2 Les particularités de MATLAB

MATLAB permet de travailler en mode interactif ou en mode programmation, tout en ayant toujours la possibilité de faire des visualisations graphiques. Considéré comme un des meilleurs langages de programmation, MATLAB possède les particularités suivantes :

- La programmation facile,
- La gamme étendue des nombres et leurs précisions,
- La bibliothèque mathématique très compréhensive,
- L'outil graphique qui inclut les fonctions d'interface graphique et les utilitaires,
- Interfaces avec C/C++, Java, .NET, Python, SQL, Hadoop et Microsoft Excel

Bibliographie

- [1] Ian F AKYILDIZ et al. « Wireless sensor networks: a survey ». In : *Computer networks* 38.4 (2002), p. 393–422.
- [2] Th ARAMPATZIS, John LYGEROS et Stamatis MANESIS. « A survey of applications of wireless sensors and wireless sensor networks ». In : *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005*. IEEE. 2005, p. 719–724.
- [3] Surthineni ASHOK et RV KRISHNAIAH. « Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology ». In : *International Journal* 3.9 (2013).
- [4] Nor Azlina Ab. AZIZ, Kamarulzaman Ab. AZIZ et Wan Zakiah Wan ISMAIL. « Coverage Strategies for Wireless Sensor Networks ». In : 3.2 (2009), p. 187–6.
- [5] Cynthia BARNHART et al. « Branch-and-price: Column generation for solving huge integer programs ». In : *Operations research* 46.3 (1998), p. 316–329.
- [6] Richard BELLMAN et E LEE. « History and development of dynamic programming ». In : *IEEE Control Systems Magazine* 4.4 (1984), p. 24–28.
- [7] Ali BENZERBADJ. « Cross-layer approach for energy efficiency and reliability in Wireless Sensor Networks dedicated to Critical Applications of Surveillance ». Theses. Université de Bretagne occidentale - Brest, juil. 2018. URL : <https://tel.archives-ouvertes.fr/tel-01891211>.
- [8] Etienne BERTIN. « Diagrammes de Voronoi 2D et 3D, applications en analyse d'images ». Thèse de doct. Université Joseph-Fourier-Grenoble I, 1994.
- [9] Mohamed Ekbal BOUZGARROU. « Parallélisation de la méthode du " Branch and Cut " pour résoudre le problème du voyageur de commerce ». Thèse de doct. Institut National Polytechnique de Grenoble-INPG, 1998.

- [10] Stephen BOYD et Lieven VANDENBERGHE. *Convex optimization*. Cambridge university press, 2004.
- [11] Chiara BURATTI et al. « An overview on wireless sensor networks technology and evolution ». In : *Sensors* 9.9 (2009), p. 6869–6896.
- [12] Samuel BURER et Adam N LETCHFORD. « Non-convex mixed-integer non-linear programming: A survey ». In : *Surveys in Operations Research and Management Science* 17.2 (2012), p. 97–106.
- [13] Vladimír ČERNÝ. « Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm ». In : *Journal of optimization theory and applications* 45.1 (1985), p. 41–51.
- [14] Yong Soo CHO et al. *MIMO-OFDM wireless communications with MATLAB*. John Wiley & Sons, 2010.
- [15] Rezaul Alam CHOWDHURY et Vijaya RAMACHANDRAN. « Cache-oblivious dynamic programming ». In : *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial et Applied Mathematics. 2006, p. 591–600.
- [16] Guoyong DAI et al. « A novel coverage holes discovery algorithm based on voronoi diagram in wireless sensor networks ». In : *International Journal of Hybrid Information Technology* 9.3 (2016), p. 273–282.
- [17] D. S. DEIF et Y. GADALLAH. « Classification of Wireless Sensor Networks Deployment Techniques ». In : *IEEE Communications Surveys Tutorials* 16.2 (2014), p. 834–855. ISSN : 1553-877X. DOI : [10.1109/SURV.2013.091213.00018](https://doi.org/10.1109/SURV.2013.091213.00018).
- [18] Robert DESIMONE, Betsy Mendieta BRITO et John BASTON. « Model of indoor signal propagation using log-normal shadowing ». In : *2015 Long Island Systems, Applications and Technology*. IEEE. 2015, p. 1–4.
- [19] Tamer M DEYAB, Uthman BAROUDI et Shokri Z SELIM. « Optimal placement of heterogeneous wireless sensor and relay nodes ». In : *2011 7th International Wireless Communications and Mobile Computing Conference*. IEEE. 2011, p. 65–70.
- [20] Marco DORIGO et Christian BLUM. « Ant colony optimization theory: A survey ». In : *Theoretical computer science* 344.2-3 (2005), p. 243–278.

- [21] Sourour ELLOUMI et al. « Optimization of wireless sensor networks deployment with coverage and connectivity constraints ». In : *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE. 2017, p. 0336–0341.
- [22] Muhammad Omer FAROOQ et Thomas KUNZ. « Operating systems for wireless sensor networks: A survey ». In : *Sensors* 11.6 (2011), p. 5900–5930.
- [23] Robert W FLOYD. « Algorithm 97: shortest path ». In : *Communications of the ACM* 5.6 (1962), p. 345.
- [24] Arnaud FRÉVILLE. « The multidimensional 0–1 knapsack problem: An overview ». In : *European Journal of Operational Research* 155.1 (2004), p. 1–21.
- [25] Fred GLOVER. « Tabu search—part I ». In : *ORSA Journal on computing* 1.3 (1989), p. 190–206.
- [26] Bhupesh Kumar GUPTA et Sudha CHANDRAKAR. « A survey on node placement for wireless sensor network using Pso algorithm ». In : *Proceedings of International Conference on Advances in Engineering & Technology*. 2014, p. 28–33.
- [27] Peter E HART, Nils J NILSSON et Bertram RAPHAEL. « A formal basis for the heuristic determination of minimum cost paths ». In : *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), p. 100–107.
- [28] Masaharu HATA. « Empirical formula for propagation loss in land mobile radio services ». In : *IEEE transactions on Vehicular Technology* 29.3 (1980), p. 317–325.
- [29] Karel HEURTEFEUX. « Protocoles localisés pour réseaux de capteurs ». Thèse de doct. INSA de Lyon, 2009.
- [30] Inaam Abbas HIEDER. « Compared to wireless deployment in areas with different environmentse ». In : *International Journal of Electrical and Computer Engineering* 9.2 (2019), p. 934.
- [31] Stefan HOUGARDY. « The Floyd–Warshall algorithm on graphs with negative cycles ». In : *Information Processing Letters* 110.8-9 (2010), p. 279–281.
- [32] Sophie JACQUIN. « Hybridation des métaheuristiques et de la programmation dynamique pour les problèmes d’optimisation mono et multi-objectif: application à la production d’énergie ». Thèse de doct. Lille 1, 2015.
- [33] Mehdi JAOUA. « Algorithme de recherche tabou pour la planification optimale d’une campagne marketing sur les moteurs de recherche ». Thèse de doct. École Polytechnique de Montréal, 2014.

- [34] Oday JEREW, Kim L BLACKMORE et al. « Estimation of hop count in multi-hop wireless sensor networks with arbitrary node density. » In : *IJWMC* 7.3 (2014), p. 207–216.
- [35] AC KAKAS et al. *Encyclopedia of Machine Learning*. 2012.
- [36] Bouabdellah KECHAR. « Problématique de la consommation de l'énergie dans les réseaux de capteurs sans fil ». In : *these de doctorat en informatique, université d'Oran* (2010).
- [37] Ines KHOUFI et al. « Survey of Deployment Algorithms in Wireless Sensor Networks: Coverage and Connectivity Issues and Challenges ». In : *International Journal of Autonomous and Adaptive Communications Systems (IJAACS)* 10.4 (2017), p. 341–390. DOI : [10.1504/IJAACS.2017.10009671](https://doi.org/10.1504/IJAACS.2017.10009671). URL : <https://hal.inria.fr/hal-01095749>.
- [38] Scott KIRKPATRICK, C Daniel GELATT et Mario P VECCHI. « Optimization by simulated annealing ». In : *science* 220.4598 (1983), p. 671–680.
- [39] Philippe LABORIE et al. « IBM ILOG CP optimizer for scheduling ». In : *Constraints* 23.2 (avr. 2018), p. 210–250. ISSN : 1572-9354. DOI : [10.1007/s10601-018-9281-x](https://doi.org/10.1007/s10601-018-9281-x). URL : <https://doi.org/10.1007/s10601-018-9281-x>.
- [40] Benjamin LÉVÊQUE. « Coloration de graphes: structures et algorithmes ». Thèse de doct. Université Joseph-Fourier-Grenoble I, 2007.
- [41] Lemia LOUAIL. « Approches cross-layer pour l'optimisation de la latence des communications dans les réseaux de capteurs sans fil ». Theses. Université de Franche-Comté, déc. 2016. URL : <https://tel.archives-ouvertes.fr/tel-01661567>.
- [42] Renaud MASSON et al. « A branch-and-cut-and-price approach for the pickup and delivery problem with shuttle routes ». In : *European Journal of Operational Research* 236.3 (2014), p. 849–862.
- [43] Kais MEKKI et al. « A comparative study of LPWAN technologies for large-scale IoT deployment ». In : *ICT Express* 5.1 (2019), p. 1–7.
- [44] Diery NGOM. « Optimizing network lifetime in wireless sensor network under coverage and network connectivity constraints ». Theses. Université de Haute Alsace - Mulhouse, mai 2016. URL : <https://tel.archives-ouvertes.fr/tel-01531464>.

- [45] Yoshihisa OKUMURA. « Field strength and its variability in VHF and UHF land-mobile radio service ». In : *Rev. Electr. Commun. Lab.* 16 (1968), p. 825–873.
- [46] *OMNeT++*. URL : <http://www.omnetpp.org>.
- [47] Theodore S RAPPAPORT et al. *Wireless communications: principles and practice*. T. 2. prentice hall PTR New Jersey, 1996.
- [48] Priyanka RAWAT et al. « Wireless sensor networks: a survey on recent developments and potential synergies ». In : *The Journal of supercomputing* 68.1 (2014), p. 1–48.
- [49] Maher REBAI et al. « Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks ». In : *Computers & Operations Research* 59 (2015), p. 11–21.
- [50] Michel SAKAROVITCH. *Optimisation combinatoire: Programmation discrète*. T. 2. Editions Hermann, 1984.
- [51] Saunhita SAPRE et S MINI. « Optimized relay nodes positioning to achieve full connectivity in wireless sensor networks ». In : *Wireless Personal Communications* (2018), p. 1–20.
- [52] Mehdi Rouan SERIK. « Application de métaheuristiques hybrides pour l’optimisation multiobjectif dans les réseaux de capteurs sans-fils ». Thèse de doct. Université d’Oran1-Ahmed Ben Bella, 2018.
- [53] FELLAH SOUMAYA. « Optimisation Multi-objectif appliquée au déploiement et à la performance des réseaux de capteurs sans fil ». Thèse de doct. Université d’Oran1-Ahmed Ben Bella, avr. 2018.
- [54] El-ghazali TALBI. « Métaheuristiques pour l’optimisation combinatoire multi-objectif: Etat de l’art ». In : *Rapport CNET (France Telecom) Octobre* (1999).
- [55] D. TAO et T. WU. « A Survey on Barrier Coverage Problem in Directional Sensor Networks ». In : *IEEE Sensors Journal* 15.2 (2015), p. 876–885. ISSN : 1530-437X. DOI : [10.1109/JSEN.2014.2310180](https://doi.org/10.1109/JSEN.2014.2310180).
- [56] Jingwen TIAN, Meijuan GAO et Guangshuang GE. « Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm ». In : *EURASIP Journal on Wireless Communications and Networking* 2016.1 (2016), p. 104.
- [57] Yacine YOUNES. « Minimisation d’énergie dans un réseau de capteurs ». Mém.de mast. UNIVERSITE MOULOUDE MAMMERI DE TIZI-OUZOU, sept. 2012.

- [58] Mohamed YOUNIS et Kemal AKKAYA. « Strategies and Techniques for Node Placement in Wireless Sensor Networks: A Survey ». In : *Ad Hoc Netw.* 6.4 (juin 2008), p. 621–655. ISSN : 1570-8705. DOI : [10.1016/j.adhoc.2007.05.003](https://doi.org/10.1016/j.adhoc.2007.05.003). URL : <http://dx.doi.org/10.1016/j.adhoc.2007.05.003>.
- [59] Wei ZENG et Richard L CHURCH. « Finding shortest paths on real road networks: the case for A ». In : *International journal of geographical information science* 23.4 (2009), p. 531–543.