

Anas EL HAOUFI

*Développement d'un modèle d'optimisation
« Location-Routing »*

Rapport de stage de 2^{ème} année

Parcours : PARCOURS INGÉNIERIE ET SCIENCES DES DONNÉES, INFORMATION, SYSTÈMES

Enseignant référent : M. Cyril PRISSETTE

Tuteur de stage : M. Pierre THIRIET

Année Universitaire 2022/2023

Engagement de non plagiat.

Je soussigné, M Anas EL HAOUFI

N° carte d'étudiant : 22107009

Déclare avoir pris connaissance de la charte des examens et notamment du paragraphe spécifique au plagiat.

Je suis pleinement conscient(e) que la copie intégrale sans citation ni référence de documents ou d'une partie de document publiés sous quelques formes que ce soit (ouvrages, publications, rapports d'étudiants, internet, etc....) est un plagiat et constitue une violation des droits d'auteur ainsi qu'une fraude caractérisée.

En conséquence, je m'engage à citer toutes les sources que j'ai utilisées pour produire et écrire ce document.

Fait le 21/08/2023

Anas EL HAOUFI

Signature(s)

Ce document doit être inséré en première page de tous les rapports, dossiers et/ou mémoires.

Remerciements

Je tiens à exprimer ma profonde gratitude envers mes encadrants, Pierre THIRIET et Thierry BIOTEAU, pour leur précieuse guidance, leurs conseils et leur soutien tout au long de ce stage. Leurs connaissances et leur disponibilité à répondre à mes questions ont grandement enrichi mon expérience. Je leur souhaite le meilleur pour tous les projets innovants qu'ils mènent au quotidien.

Je souhaite également adresser mes remerciements sincères à l'équipe pédagogique de l'école pour leur engagement envers notre formation et leur contribution à notre apprentissage. Leurs efforts pour fournir un environnement propice à l'apprentissage et à la croissance personnelle ont été inestimables.

Un remerciement spécial va à ma famille, en particulier à mon grand frère, pour leur soutien tout au long de mon parcours. Leur encouragement constant et m'a permis d'atteindre chaque étape de cette aventure.

Enfin, je tiens à exprimer ma reconnaissance pour toutes les personnes qui ont contribué de près ou de loin à la réussite de ce stage. Votre soutien et vos encouragements ont joué un rôle déterminant dans cette expérience enrichissante.

Merci à tous.

AVANT-PROPOS

Pour débiter ce rapport de stage, il convient de présenter tout d'abord l'environnement de travail où s'est déroulé ce travail. L'organisme d'accueil est l'INRAE, institut résultant de la fusion entre l'INRA (Institut National de la Recherche Agronomique) et l'IRSTEA (Institut national de Recherche en Sciences et Technologies pour l'Environnement et l'Agriculture) en 2020.

L'INRAE a pour mission de réaliser, organiser et coordonner des travaux de recherches scientifiques et technologiques dans divers domaines tels que l'agriculture, l'alimentation, la forêt, l'environnement, l'eau, la biodiversité, la bioéconomie, l'économie circulaire, la gestion durable des territoires et des risques.

Au sein de cet institut de recherche national, l'unité OPAALE se situe dans le quartier Beauregard de Rennes. Cette unité est divisée en quatre équipes travaillant sur des sujets variés, dont l'équipe SAFIR qui se concentre sur les filières de valorisation des résidus organiques. C'est au sein de cette équipe que le projet PAPscale a été construit, et c'est dans le cadre de ce projet que s'inscrit mon stage. La Figure 1 représente l'organigramme du stage.

Figure 1 : représentation de l'organigramme du stage

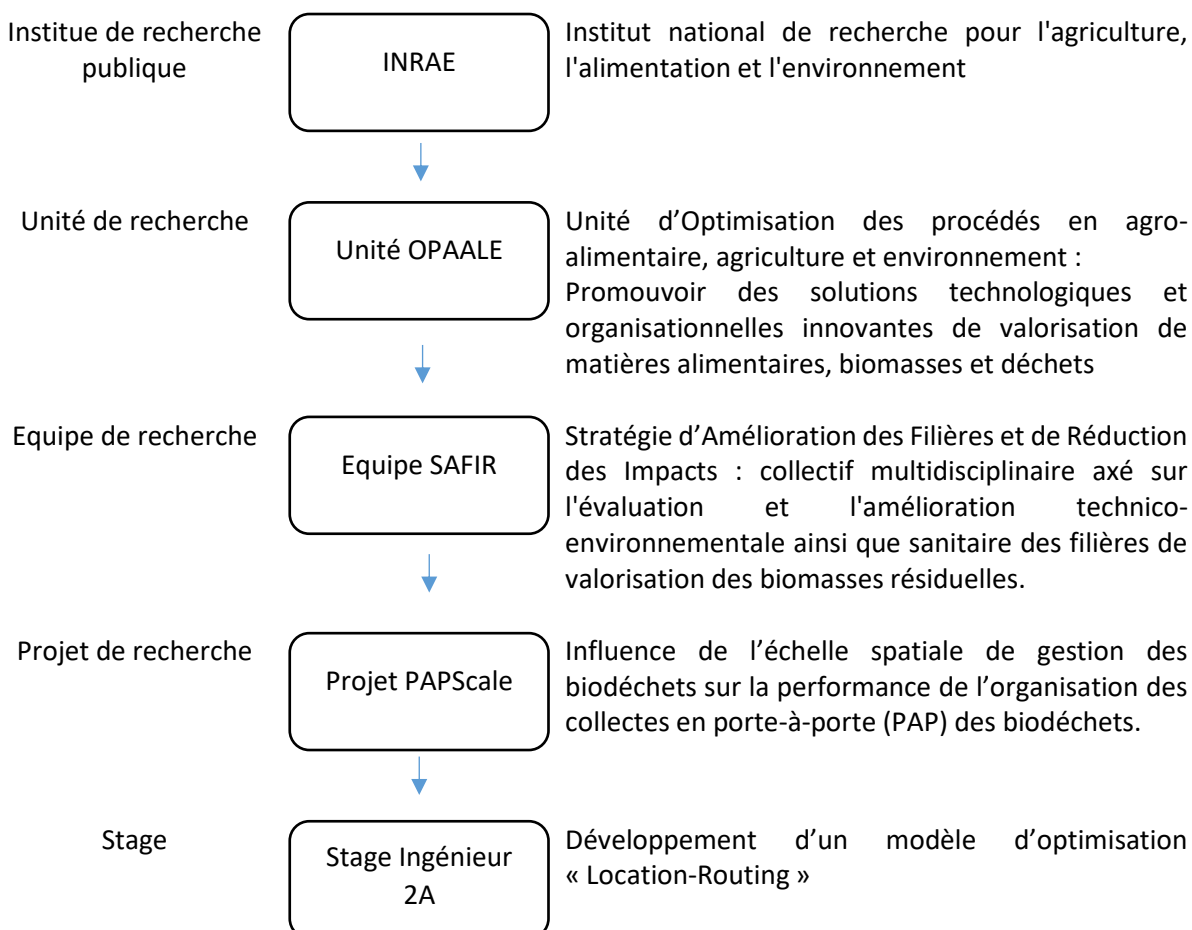


TABLE DES MATIERES

Remerciements	3
AVANT-PROPOS.....	4
Introduction.....	6
1. Contexte de gestion des biodéchets en France	7
1.1. Gestion et collecte des biodéchets en France.....	7
2. Sujet de stage	7
2.1. Projet PAPscale.....	7
2.2. Objectifs du stage	8
3. Problème de Location-routing	8
3.1. Variantes du Location-Routing Problem	9
3.2. Méthode de résolution du problème.....	9
4. Méthode.....	10
4.2. Formulation mathématique	10
4.3.1. <i>Approche de résolution</i>	13
5. Résultats.....	19
5.1. Description des instances.....	19
5.2. Configuration des paramètres.....	20
5.3. Résultat du modèle	21
5.4. Améliorations de l'implémentation du modèle	21
6. Bilan du stage	22
6.1. Réflexions sur les avancées	22
7. Conclusion	24
8. Bibliographie.....	25
9. Annexe 1.....	26
9.1. Graphe représentatif des solutions et variation des coûts	26
9.2. Scores des opérateurs et informations générales.....	27

Introduction

Les déchets ménagers et assimilés englobent les résidus issus des activités quotidiennes des foyers, principalement dans des contextes résidentiels. Ils comprennent une diversité de matériaux tels que les restes alimentaires, les emballages en carton, plastique et verre, les papiers, ainsi que d'autres articles indésirables provenant des ménages. Parmi ces déchets, les éléments d'origine organique, tels que les restes alimentaires, également connus sous le nom de déchets de cuisine et de table (DCT), ou les déchets verts issus de l'entretien d'un jardin, sont qualifiés de "biodéchets".

Les biodéchets constituent une part considérable des déchets ménagers et assimilés (DMA), représentant environ 38% des déchets éliminés dans les ordures ménagères résiduelles (OMR) [4, p. 10]. Ce constat soulève un problème majeur : en l'absence de traitement approprié, ces déchets posent des risques pour l'environnement en termes de coûts et de pollution. Parallèlement, En les considérant comme de simples déchets, nous perdons l'opportunité précieuse de les valoriser en produisant du compost ou de l'énergie, contribuant à une économie plus verte et à la création de nouvelles filières économiques.

La gestion des biodéchets présente plusieurs défis, parmi lesquels l'enjeu de la collecte. Cette phase cruciale implique la récupération des déchets produits par les ménages, les entreprises et autres sources, en vue de leur acheminement vers les installations de traitement appropriées. Une collecte efficace des biodéchets, tels que les déchets de cuisine et de table, joue un rôle clé dans la réduction des émissions de gaz à effet de serre, tout en minimisant les pertes financières liées aux coûts de transport[1]. Parmi les approches de collecte, nous nous concentrons ici sur la méthode en porte-à-porte (PAP) en raison de ses performances remarquables en termes de quantité et de qualité des déchets collectés.

Actuellement, la gestion et la collecte des biodéchets sont organisées à l'échelle administrative des Établissements publics de coopération intercommunales (EPCI). Cependant, cette approche soulève des questions concernant l'efficacité de ce système de traitement des biodéchets et suscite l'intérêt pour explorer d'autres systèmes de collecte basés sur des échelles différentes. Cela souligne la nécessité de comparer l'efficacité des systèmes de collecte à différentes échelles, en évaluant les performances de systèmes avec des choix variés de positions pour les sites de traitement.

Le choix des positions des sites de traitement et des itinéraires associés constitue un défi bien connu en logistique, désigné sous le nom de Location-routing Problem (LRP)[2]. Il s'agit d'un problème d'optimisation complexe, que des agents de l'équipe SAFIR de l'Unité OPAALE ont abordés à travers des projets antérieurs mais de manière non exhaustive, sans être sûrs que les outils testés répondaient pleinement à leurs objets d'études.

C'est précisément là que j'interviens en tant que stagiaire au sein de l'INRAE, pour contribuer à identifier la solution adaptée. Mon objectif est d'agrégier les connaissances disponibles et pertinentes pour le contexte étudié en examinant la littérature spécialisée puis de proposer des solutions dédiées pour résoudre efficacement le problème posé.

1. Contexte de gestion des biodéchets en France

1.1. Gestion et collecte des biodéchets en France

Depuis 2015, la loi de Transition Énergétique pour la Croissance Verte (LTECV) et la loi Anti-Gaspillage pour une Économie Circulaire (AGEC) de 2020 imposent aux collectivités l'obligation de proposer des solutions de tri à la source des biodéchets à tous les producteurs, et de déployer ces solutions avant le 31 décembre 2023. Certaines collectivités ont déjà pris des mesures en investissant dans des collectes de biodéchets ou en distribuant des composteurs pour une utilisation à domicile. En 2020, d'après l'ADEME (Déchets chiffres clés, 2020), les biodéchets représentent environ 38% des déchets jetés dans les poubelles OMR (Ordures Ménagères Résiduelles)[4, p. 10]. Ces évolutions réglementaires ont suscité un intérêt croissant pour l'étude des différentes modes de collecte de déchets.

Suivant les déchets que nous produisons plusieurs modes de collecte sont possibles. Ces modes diffèrent selon la nature du déchet mais chaque collectivité peut adopter le mode de tri et le mode de collecte qu'elle identifie comme le plus adapté à son territoire. En effet selon le contexte géographique de la zone concernée, sa densité de population ou son type d'urbanisation, le geste de tri peut varier. Cette variété de pratique de tri peut se rencontrer entre territoires mais également au sein d'un même territoire.

1.1.1. Collecte porte-à-porte

Parmi les modes de collecte utilisés se trouve la porte à porte (PAP). Il consiste à disposer ses déchets devant chez soi ou au sein d'une partie commune pour qu'un véhicule les collectent lors d'une tournée dédiée à ce type de déchet. Cette collecte nécessite un planning de collecte que le producteur devra suivre pour mettre à disposition sa poubelle à la collecte. Ce type de collecte est coûteux pour la collectivité tant financièrement qu'en terme de bilan carbone, néanmoins il s'avère très efficace en termes de quantité et qualité de flux.

1.1.2. Point d'apport volontaire

Un autre mode de collecte nommé apport volontaire consiste en un ou plusieurs bacs ou containers où chaque personne est libre d'amener son déchet. Il est soumis à une forte problématique spatiale puisque la localisation de ces PAV doit répondre à une distance raisonnable pour tout usager, et doit également concerner un nombre maximal de personnes selon la contenance du point d'apport, l'enjeu de son emplacement est central pour ce type de collecte. La collecte est ensuite effectuée en véhicule. Le mode de collecte en PAV s'avère beaucoup moins coûteux que le porte à porte, mais la quantité de déchets collectée est en général moindre pour une population desservie égale.

2. Sujet de stage

Les biodéchets constituent une part significative des déchets ménagers et assimilés (DMA), et leur collecte porte-à-porte (PAP) est largement adoptée pour assurer une gestion efficace. En optimisant le processus de collecte PAP, nous pouvons véritablement maximiser la valorisation des biodéchets.

2.1. Projet PAPscale

La gestion des DMA est ainsi pensée et optimisée dans les limites géographiques de chaque EPCI. Or ces limites sont issues de processus historiques liés à des enjeux plus politiques que fonctionnels. Elles ne sont donc pas nécessairement adaptées aux besoins spécifiques de la gestion des DMA.

C'est là que le projet PAPscale de l'INRAE intervient, en questionnant l'échelle spatiale de gestion des DMA et en particulier des biodéchets et en vérifiant deux hypothèses clés :

- L'échelle spatiale de gestion des biodéchets a un impact significatif sur la performance de l'organisation de la collecte PAP des biodéchets.

- L'échelle fonctionnelle actuelle, basée sur l'EPCI, n'est pas forcément la plus optimale, et d'autres organisations résultant de collaborations interterritoriales pourraient s'avérer mieux adaptées.

Ce travail nécessite de comparer des systèmes de collecte PAP à différentes échelles spatiales en termes de performances. Pour ce faire, le projet adoptera une approche originale pour la conception de systèmes optimisés à différentes échelles. Elle intégrera les questions de trajets de collecte, mais aussi d'emplacements des nouvelles infrastructures de transfert et de valorisation si nécessaire.

2.2. Objectifs du stage

La comparaison des systèmes de collecte PAP à différentes échelles nécessite d'élaborer des modèles de logistiques réalistes qui incluent les trajets de collectes et l'emplacement géographique de sites de valorisation des biodéchets. Cette approche repose sur le développement d'un modèle d'optimisation appliqué à différentes échelle de gestion spatiale afin d'analyser les performances de chaque système, en prenant en compte des facteurs tels que les distances des trajets et les coûts de fonctionnement des sites de traitement.

Dans le cadre de ce stage, mon rôle consiste à participer au développement de ce modèle en utilisant une approche de type Location-Routing Problem (LRP). Ce modèle intégrera de manière conjointe les éléments clés de la filière, en se concentrant à la fois sur le transport (la collecte des déchets) et sur la localisation des infrastructures (les sites de traitement).

Le processus sera réalisé en quatre étapes principales :

- Identification des différentes formulations de modèle de « Location–Routing » (objectifs, contraintes, etc.) par un travail bibliographique.
- Proposition d'une formulation du problème et identification d'une solution adaptée au besoin du projet.
- Développement d'un modèle informatique d'optimisation en langage de programmation Python. Une alternative pourrait éventuellement utiliser le langage Julia pour ses performances en termes de temps d'exécution.

3. Problème de Location-routing

Le Location-Routing Problem (LRP) est un problème d'optimisation complexe qui combine deux aspects essentiels de la logistique : la localisation des installations et la planification des itinéraires de livraison. Auparavant, ces deux problèmes d'optimisation combinatoire ont été résolus séparément en déterminant d'abord la localisation des installations, puis en planifiant les itinéraires par la suite. Ces approches traditionnelles conduisent à un résultat sous-optimal, étant donné qu'elles sont étroitement liées entre elles (Salhi and Rand, 1989 ; Salhi and Nagy, 1999).

Dans notre contexte, le LRP devient particulièrement pertinent car il permet d'optimiser à la fois l'emplacement des sites de traitement des biodéchets et les itinéraires des camions de collecte. L'objectif est de minimiser les coûts opérationnels, comme les distances parcourues par les camions de collecte, tout en répondant aux contraintes spécifiques du problème, comme le nombre de camions de collecte dans chaque site de traitements.

Afin de faciliter la description du problème, il est utile d'utiliser les termes "dépôts" et "clients" pour faire référence, respectivement, aux sites de traitement des biodéchets et aux points de génération des biodéchets.

Les entrées du modèle LRP comprennent généralement les éléments suivants :

- Des nœuds : information sur les dépôts potentiels à ouvrir et clients,
- Une matrice des coûts de déplacements entre deux nœuds,
- Coût de l'ouverture d'un dépôt,
- Coût de l'utilisation d'un véhicule,
- Capacité de chaque véhicule,
- Nombre de véhicules sur chaque dépôt.

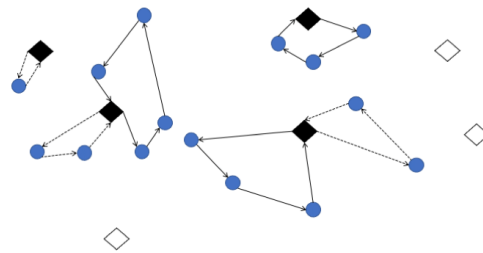


Figure 2 planning des trajets, avec les clients en bleu, les dépôts ouverts en noir, et les dépôts fermés en blanc.
Source : D. Tordecilla 2021

Le modèle donne en sortie un planning optimisé contenant : les dépôt à ouvrir en noir sur Figure 2, les itinéraire à prendre par chaque véhicule en trait noir, et le coût total du planning (coût des trajets des camions, coût des ouvertures des dépôts et de l'utilisation d'un véhicule). Le but c'est de minimiser ce dernier.

Remarque : On peut dans certains cas choisir un nombre à ne pas dépasser de dépôts à ouvrir et de véhicules à utiliser.

3.1. Variantes du Location-Routing Problem

Le problème du Location-Routing Problem (LRP) est un problème d'optimisation combinatoire qui présente plusieurs variantes, chacune avec ses caractéristiques et ses contraintes spécifiques. Voici quelques-unes des variantes courantes du LRP :

LRP avec capacité des véhicules : Dans cette variante, les véhicules de collecte ont une capacité limitée en termes de poids, de volume ou de nombre de conteneurs qu'ils peuvent transporter. L'objectif est de déterminer les itinéraires de collecte et les emplacements des dépôts tout en respectant les contraintes de capacité des véhicules.

LRP with Time Window : Cette variante prend en compte des contraintes de temps pour la collecte des clients, ce qui signifie que les véhicules doivent respecter des fenêtres de temps spécifiques pour la collecte. Les itinéraires doivent être planifiés de manière à satisfaire les contraintes de temps tout en minimisant les distances parcourues.

LRP avec plusieurs types de véhicules : Dans cette variante, il y a plusieurs types de véhicules de collecte, chacun ayant ses propres caractéristiques et capacités. L'objectif est de déterminer quelle combinaison de types de véhicules et d'itinéraires est la plus efficace pour la collecte des clients.

LRP périodique : Cette variante du LRP implique une collecte répétée à des intervalles réguliers, par exemple, quotidiennement, hebdomadairement ou mensuellement. L'objectif est de planifier les itinéraires de collecte sur une période donnée, en tenant compte des contraintes de capacité des véhicules et des fenêtres de temps.

LRP multi-objectif : Cette variante du LRP considère plusieurs objectifs simultanément, tels que la minimisation des coûts de collecte, la réduction des émissions de gaz à effet de serre et la satisfaction des clients. L'objectif est de trouver un compromis entre ces objectifs souvent contradictoires.

LRP stochastique : Dans cette variante, certaines des données du problème, telles que les demandes des clients ou les temps de service, sont soumises à l'incertitude. L'objectif est de trouver des solutions robustes qui sont efficaces dans un large éventail de scénarios possibles.

3.2. Méthode de résolution du problème

Dans le contexte du Location-Routing Problem (LRP) ou de tout autre problème d'optimisation, la résolution consiste à trouver les meilleures valeurs des variables du modèle mathématique pour atteindre l'objectif défini tout en respectant les contraintes. Il est important de noter que de nombreux

problèmes d'optimisation, y compris le LRP, sont considérés comme des problèmes NP-difficiles, ce qui signifie qu'il n'existe pas d'algorithme polynomial connu pour les résoudre de manière exacte dans un temps raisonnable.

Pour résoudre un problème d'optimisation comme le LRP, plusieurs approches peuvent être utilisées.

3.2.1. Méthodes exactes

Ces méthodes garantissent de trouver la meilleure solution possible, mais elles peuvent être très coûteuses en temps de calcul, surtout pour des problèmes de grande taille. Parmi les méthodes exactes, on trouve la programmation linéaire, la programmation linéaire en nombres entiers (PLNE), et la programmation par contraintes (PPC).

3.2.2. Méthodes heuristiques

Ces méthodes sont des approches plus rapides qui ne garantissent pas de trouver la meilleure solution, mais elles donnent souvent des résultats de bonne qualité en un temps de calcul raisonnable. Les heuristiques utilisent des règles et des stratégies spécifiques pour explorer l'espace des solutions de manière plus efficace. Par exemple, on peut utiliser des métaheuristiques comme les algorithmes génétiques (GA), le Large Neighborhood Search (LNS), simulated annealing (SA), etc.

4. Méthode

4.1. Type du LRP et choix de la méthode de la résolution

Dans le cadre du projet PAPscale, l'objectif de l'élaboration du modèle du LRP est de générer une planification optimale du transport des biodéchets tout en prenant en compte les contraintes liées à la capacité des camions et des dépôts. Cependant, pour des raisons de simplification, le modèle ne prend pas en compte certaines contraintes supplémentaires, telles que les contraintes de temps de collecte ou les distances parcourues par chaque camion. Prenant en compte les capacités des véhicules et des dépôts, notre problème s'inscrit dans la catégorie du Location-Routing Problem avec capacité de véhicule (CLRP)

Notre problème impliquera des systèmes composés de milliers de points à différentes échelles. Cependant, nous sommes confrontés à une contrainte de temps qui nous empêche d'utiliser des méthodes exactes. Par conséquent, compte tenu des objectifs spécifiques du projet, l'obtention de la solution optimale n'est pas nécessaire ; une solution offrant un coût satisfaisant peut-être suffisante. Dans ce contexte, les méthodes heuristiques se révèlent être le choix le plus pertinent pour résoudre efficacement notre problème du CLRP. Ces approches plus rapides et adaptatives nous permettront d'obtenir des solutions de qualité en un temps de calcul raisonnable.

4.2. Formulation mathématique

Le LRP peut être défini sur un graphe complet, pondéré et non orienté $G(V, A, C)$, dans lequel V représente l'ensemble des nœuds (comprenant le sous-ensemble J des emplacements potentiels de dépôts et le sous-ensemble I des clients), A représente l'ensemble des arcs, et C est la matrice des coûts de traversée de chaque arc.

Un ensemble de véhicules homogènes illimités avec des contraintes de capacité (K) est disponible pour effectuer les itinéraires. De plus, on suppose que tous les véhicules sont partagés par tous les dépôts (c'est-à-dire qu'aucun dépôt n'a une flotte spécifique) et que chaque arc $a \in A$ satisfait l'inégalité triangulaire. Les demandes des clients sont déterministes et connues à l'avance. Chaque client doit être desservi par un seul véhicule depuis le dépôt qui lui a été attribué. Les contraintes suivantes doivent être satisfaites : (a) le nombre total des clients attribués à un dépôt ne doit pas dépasser sa capacité, (b) chaque itinéraire commence et se termine au même dépôt, (c) chaque véhicule effectue au plus un trajet, (d) chaque client est desservi par un seul véhicule, et (e) le nombre

total des clients visités par un véhicule correspond à sa capacité. Notre CLRP peut être formulé comme un modèle de programmation mathématique, dont les ensembles, les paramètres et les variables sont présentés dans le tableau 1.

Tableau 1

<i>Sets</i>	
V	<i>Set of nodes</i>
K	<i>Set of vehicles</i>
I	<i>Set of customers, $I \subset V$</i>
J	<i>Set of depots, $J \subset V$</i>
A	<i>Set of arcs, $A = V \times V = \{(m, n) : m \in V, n \in V \wedge m \neq n\}$</i>
$\delta^+(S)$	<i>Set of arcs leaving S, $S \subset V, \delta^+(S) \subset A$</i>
$\delta^-(S)$	<i>Set of arcs entering S, $S \subset V, \delta^-(S) \subset A$</i>

<i>Parameters</i>	
f_j	<i>Fixed opening cost of depot $j \in J$</i>
o_j	<i>Variable opening cost of depot $j \in J$</i>
c_a	<i>Cost of traversing arc $a \in A$</i>
q	<i>Capacity of each vehicle</i>
s	<i>maximum number of vehicul per depot</i>
M	<i>A very large number when compared to the magnitude of the rest of the parameters</i>

<i>Variables</i>	
y_j	<i>Binary variable equal to 1 if depot $j \in J$ is open, 0 otherwise</i>
x_{ij}	<i>Binary variable equal to 1 if customer $i \in I$ is assigned to depot $j \in J$, 0 otherwise</i>
w_{ak}	<i>Binary variable equal to 1 if arc $a \in A$ is used in the route performed by vehicle $k \in K$, 0 otherwise</i>
o_{jk}	<i>Binary variable equal to 1 if vehicle k is used by depot j</i>
u_{ik}	<i>visited client by vehicle $k \in K$ until customer $i \in I$</i>

Le but est de minimiser la fonction qui représente les coûts totaux :

$$\text{Minimize } \sum_{j \in J} f_j y_j + \sum_{a \in A} \sum_{k \in K} c_a w_{ak} \quad (1)$$

Contraints :

$$\sum_{a \in \delta^-(i)} \sum_{k \in K} w_{ak} = 1, \quad \forall i \in I \quad (2)$$

$$\sum_{i \in I} \sum_{a \in \delta^-(i)} w_{ak} \leq q, \quad \forall k \in K \quad (3)$$

$$\sum_{a \in \delta^+(n)} w_{ak} = \sum_{a \in \delta^-(n)} w_{ak}, \quad \forall k \in K, \forall n \in V \quad (4)$$

$$\sum_{a \in \delta^+(J)} w_{ak} \leq 1, \quad \forall k \in K \quad (5)$$

$$u_{ik} + 1 \leq u_{hk} + M(1 - w_{ak}), \quad \forall a \in \delta^+(i \in I) \cap \delta^-(h \in I), \quad \forall k \in K \quad (6)$$

$$\sum_{a \in \delta^+(j)} wak + \sum_{a \in \delta^-(i)} wak \leq 1 + x_{ij}, \quad \forall i \in I, \forall j \in J, \forall k \in K \quad (7)$$

$$\sum_{j \in J} x_{ij} = 1, \quad \forall i \in I \quad (8)$$

$$\sum_{k \in K} o_{jk} \leq s, \quad \forall j \in J \quad (9)$$

$$\forall y_j, x_{ij}, wak \in \{0,1\} \quad (10)$$

$$\forall u_{ik} \geq 0 \quad (11)$$

La fonction objectif (1) minimise le coût total, qui est lié aux coût fixe d'ouverture d'un dépôt et les coûts des trajets. La contrainte (2) garantie que chaque client est desservi par une seule route. La contrainte (3) est associée avec la capacité de véhicule. Les contraintes (4) et (5) garantie la continuité de la route et le retour vers le dépôt associé. La contrainte (6) est pour éliminer les sous-routes (Subtour). La contrainte (7) garantie qu'un client ne peut pas être associé à aucun dépôt sauf s'il n'existe pas de routes reliant clients et dépôt. La contrainte (8) assure qu'un client ne peut être associé qu'à un seul dépôt. La contrainte (9) interdit le dépôt de dépasser le nombre maximal de véhicules. Les contraintes (10) et (11) définissent les variables du problème. Néanmoins, différents modèles peuvent être formulés pour représenter le même problème.

4.3. Heuristique choisie

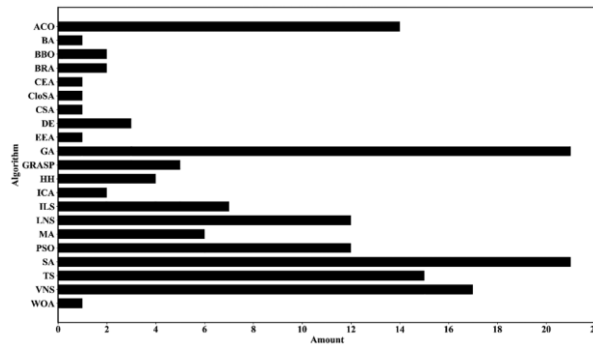


Figure 3 Location-routing problem: a classification of recent research, Mara et Al 2021 [5]

En explorant la littérature, le Simulated Annealing (SA) et les algorithmes génétiques (GA) sont les heuristiques les plus utilisées pour résoudre le LRP (voir Figure 3). Mais les travaux récents dans ce problème indiquent également l'émergence du Large Neighborhood Search [5, p. 2955], une approche d'optimisation métaheuristique qui se concentre sur l'exploration de voisinages dans l'espace des solutions. Cette méthode se montre prometteuse pour résoudre des problèmes d'optimisation complexes tels que le Location-Routing Problem (LRP), car elle permet d'explorer un ensemble plus vaste de solutions potentielles, ce qui peut conduire à de meilleures solutions en termes de qualité et d'efficacité.

Compte tenu des indices prometteurs quant à l'efficacité de l'algorithme et des ressources disponibles, le Large Neighborhood Search (LNS) se révèle être une approche plus pertinente et aisée pour résoudre notre problème.

4.3.1. Approche de résolution

Dans la section (2.3.1), nous avons identifié que la méthode la plus pertinente dans le contexte de cette étude est le Large Neighborhood Search (LNS). En examinant la littérature, nous avons également constaté que pour résoudre des problèmes du LRP, une extension appelée Adaptive Large Neighborhood Search (ALNS) est fréquemment utilisée.

L'Adaptive Large Neighborhood Search (ALNS) a été introduit par *Ropke et Pisinger* [2] pour résoudre plusieurs variantes du Vehicle Routing Problem (VRP), un problème similaire mais avec un seul dépôt. Depuis lors, cette métaheuristique et ses variations ont produit d'excellents résultats sur plusieurs variantes complexes du LRP (voir *Hemmelmayr et al., 2012 et 2015 ; Koç et al., 2016*). Dans cette partie, nous allons examiner en quoi consiste cette approche et comment elle peut être mise en œuvre pour obtenir des solutions efficaces pour notre problème.

4.3.2. Adaptive Large Neighborhood search

L'idée générale est de retirer de manière répétée des clients ou des dépôts de la solution et de les réinsérer à des positions plus avantageuses, voir Figure 4. Cela est réalisé à l'aide d'heuristiques de destruction et de réparation spéciales. Contrairement à l'heuristique de LNS proposée par Shaw[7] (Parallel Large Neighborhood Search 2003), l'ALNS utilise plusieurs heuristiques de destruction et de réparation au lieu d'une seule pour chacune. À chaque itération, une heuristique de destruction et une heuristique de réparation sont choisies et appliquées. La sélection est basée sur le succès passé des heuristiques.

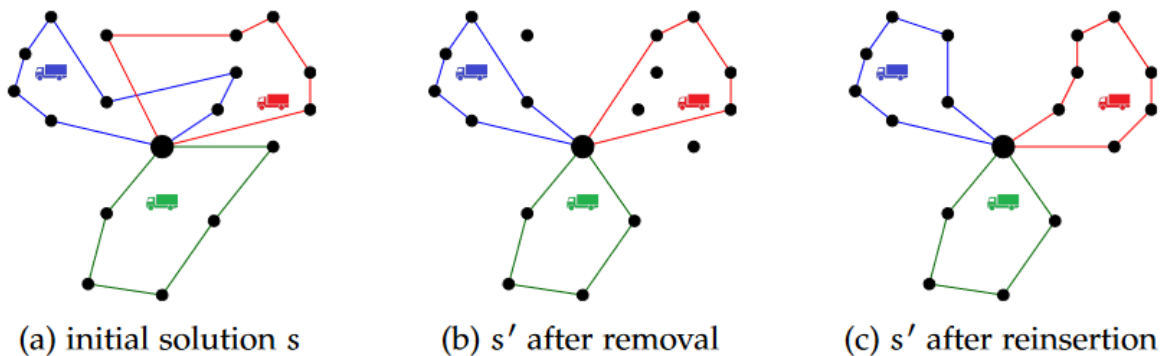


Figure 4 : Adaptive Large Neighborhood Search, Roman Lutz 2014 [8]

Nous nous sommes inspirés de l'algorithme élaboré par *Hemmelmayr*[9], qui résout un problème de LRP et de VRP à deux échelons, similaires à notre problématique. Le fonctionnement de l'algorithme, présenté dans Figure 5, est le suivant :

À partir d'une solution initiale s , avec un coût $f(s)$, un opérateur de destruction est d'abord sélectionné pour retirer q clients, puis un opérateur de réparation est utilisé pour les réinsérer dans la solution actuelle. Étant donné que nous traitons un LRP (dépôts et clients), nous utilisons une structure hiérarchique pour les opérateurs de destruction. Il existe deux types d'opérateurs de destruction : ceux qui modifient la configuration donnée des dépôts en ouvrant ou fermant des dépôts, et ceux qui n'affectent qu'une partie plus restreinte de la solution comme la suppression d'un nombre de clients ou de routes. Dans ce qui suit, DL (pour Large Destructor) représente l'ensemble des opérateurs de destruction ayant un impact sur les dépôts, DS (pour Small Destructor) l'ensemble des opérateurs qui suppriment les clients de la solution, et R est l'ensemble des opérateurs de réparation.

Algorithme 1

```

s ← InitialSolution, InitializeScores( $\pi$ ), i ← 0
repeat
  if i =  $\omega$  then
    N- ← ChooseDestroyOperator(DL,  $\pi$ )
  else
    N- ← ChooseDestroyOperator(DS,  $\pi$ )
  end if
  N+ ← ChooseRepairOperator(R,  $\pi$ )
  s' ← DestroyAndRepair(s, N-, N+)
  if i =  $\omega$  then
    s' ← LocalSearch(s')
    s ← s' // free pass
    i ← 0
  else if f(s') < (1 +  $\theta$ )f(s*) then
    s' ← LocalSearch(s')
  end if
  if f(s') < f(s) then
    s ← s'
    i ← 0
  else
    i ← i + 1
  end if
  if f(s) < f(s*) then
    s* ← s
  end if
  Update scores ( $\pi$ )
until the stopping condition is met
return s*

```

Figure 5 : Algorithme de l'ALNS, Hemmelmayr 2012.

Les opérateurs de destruction de l'ensemble DL, c'est-à-dire "Suppression de dépôt", "Ouverture de dépôt" et "Échange de dépôt", sont exécutés chaque fois que w itérations sont effectuées sans amélioration. La nouvelle solution obtenue par l'un de ces opérateurs passe ensuite par une phase de recherche locale. De plus, elle est acceptée comme nouvelle solution candidate même si elle ne présente pas d'amélioration, c'est-à-dire qu'elle bénéficie d'une exemption dans la décision d'acceptation.

Les solutions obtenues par les opérateurs de l'ensemble DS ne passent par la recherche locale que si elles ne dépassent pas la meilleure solution connue *s*^{*} d'une valeur seuil déterminé par le paramètre θ . De plus, elles ne sont acceptées comme nouvelle solution candidate que si elles ont une meilleure valeur objective que la solution candidate actuelle.

La recherche n'est pas limitée aux solutions qui respectent les contraintes (feasible). Au lieu de cela, nous autorisons les violations des contraintes liées à la capacité des véhicules, au nombre de véhicules disponibles, et nous utilisons une fonction de pénalité pondérée (Weighted Penalty Function WPN) pour prendre en compte ces violations.

4.3.2.1. Destructeurs utilisés

Dans ce qui suit, nous décrivons les opérateurs de destruction utilisés dans notre code, le choix de ces opérateurs est basé sur leurs performances au niveau du résultat et du temps de calcul au cours des essais sur différentes instances. Ils existent deux grands types d'opérateurs, les grands destructeurs DL et les petits destructeurs DS.

Close DEPOT ou Suppression de dépôt (DL) : Cet opérateur choisit aléatoirement un dépôt parmi ceux ouverts et il le ferme en le plaçant dans le pool des dépôt fermé (Unassigned Depot). Tous les clients actuellement attribués à ce satellite sont retirés et placés dans le pool de clients. De plus, pour éviter la fermeture de tous les dépôts, nous sélectionnons parmi les autres dépôts un aléatoirement et l'ouvrons, s'il n'est pas déjà ouvert.

Open DEPOT ou Ouverture de dépôt (DL) : Nous choisissons aléatoirement un dépôt parmi ceux qui sont fermés et l'ouvrons. Ensuite, les q clients les plus proches de ce dépôt sont supprimés et placés dans le pool de clients non assignés (Unassigned Clients).

Swap DEPOT ou Echange de dépôt (DL) : Pour commencer, l'opérateur supprime un dépôt de Dépôt. Ensuite, nous sélectionnons aléatoirement un nouveau dépôt en utilisant une méthode de sélection roue de roulette (Roulette Wheel¹), et nous l'ouvrons. La probabilité de choisir un dépôt est inversement proportionnelle à la distance par rapport au dépôt qui a été supprimé. Notre objectif est de privilégier les échanges de dépôts qui sont proches les uns des autres, afin de limiter les modifications drastiques de la solution.

¹ Une technique de sélection des éléments à base de leurs probabilités.

La probabilité de choisir un dépôt est $pi = \frac{1}{di \sum \frac{1}{dn}}$ avec di la distance du dépôt i par rapport au dépôt enlevé.

Proximity Removal (DS) : Cet opérateur sélectionne un groupe de points proches, les élimine et les place dans le pool des clients non assignés. Dans notre programme, nous utilisons la méthode des K-moyennes pour répartir les clients en clusters en fonction de leurs positions spatiales.

Worst Removal (DS) : Cet opérateur élimine les q clients ayant le coût de desserte le plus élevé. Plus précisément, le gain est défini comme la différence entre le coût lorsque le client est dans la solution et le coût lorsqu'il est supprimé. Dans cette méthode, les coûts sont modifiés aléatoirement par un facteur $[0.9, 1.1]$. Pour des raisons de diversification.

Random Removal (DS) : ici, nous choisissons des clients aléatoirement pour les éliminer et les mettre dans le pool des clients non assignés.

4.3.2.2. Réparateurs utilisés

Tout comme les destructeurs, le choix de ces réparateurs s'appuie sur la littérature existante ainsi que sur leurs performances démontrées dans des expériences antérieures.

Greedy Repair : Avant d'insérer un client parmi ceux du pool des clients non assignés, l'opérateur calcule les coûts d'insertion pour chaque position potentielle dans la solution, et opte pour la position présentant le coût d'insertion le plus bas en termes de distance. Comme le Worst Removal, le calcul est perturbé par un facteur aléatoire.

Regret Repair : Contrairement au Greedy Repair, cet opérateur ne choisit pas le client à insérer de manière aléatoire. Au lieu de cela, il calcule une valeur appelée Regret. Plus cette valeur est élevée, plus le risque d'une insertion coûteuse augmente si le client n'est pas inséré en premier. Le calcul est aussi perturbé pour diversifier les solutions. Voir Ropke and Pisinger 2006 pour plus de détails.

4.3.2.3. Fonction de pénalité pondérée (WPN) [3, N° 4.1]

Afin d'explorer un maximum de solutions pendant notre recherche, nous permettons des violations des contraintes de capacité des véhicules et du nombre de véhicules disponibles. Nous utilisons une fonction de pénalité pondérée pour tenir compte de ces violations. Plus précisément, nous définissons la fonction objectif $f(s) = c(s) + ad(s) + be(s)$, où $c(s)$ représente le coût opérationnel du système (comprenant le coût de trajet et le coût d'ouverture des dépôts), $d(s)$ et $e(s)$ correspondent aux violations des contraintes de capacité des véhicules et de nombre de véhicules, respectivement, et a et b sont les poids associés.

Ces poids a et b sont ajustés dynamiquement pendant la recherche dans l'intervalle $[l, k]$. Le point de départ inférieur garantit qu'en cas de violation, l'algorithme démarre avec une valeur raisonnable, tandis que le point de départ supérieur évite que les poids ne deviennent infinis. Lorsqu'une contrainte de capacité des véhicules ou du nombre de véhicules disponibles est violée, le poids correspondant est multiplié par un facteur $\delta > 1$. Lorsque la solution est réalisable, le poids associé est divisé par δ .

4.3.2.4. Sélection des opérateurs

La sélection est basée sur le succès de chaque opérateur dans les itérations précédentes. Les opérateurs de destruction et de réparation sont pondérés et choisis indépendamment. Chaque fois que l'opérateur j trouve une nouvelle meilleure solution optimal, nu est ajoutée au score p_j . Quand j trouve une solution acceptable, $nu/2$ est ajoutée au score p_j . La sélection des opérateurs est basée sur une sélection par roue de roulette.

4.3.2.5. Recherche locale

La recherche locale permet d'explorer les gains possibles obtenus à partir de petites modifications d'une solution donnée. Elle est effectuée après l'application d'un parmi les grands destructeurs DL. Elle est également exécutée quand le coût d'une solution est inférieur à $(1 + \theta)$ le coût de la solution meilleure solution trouvée. Ici, nous allons décrire chaque recherche locale utilisée.

Move 1 point : c'est la première recherche locale utilisée, déplace chaque client vers la meilleure position, en termes de coût de trajet, dans la même route ou dans une route différente.

La recherche locale 2-opt [10]: l'opérateur 2-opt fonctionne en examinant chaque paire de clients adjacents dans une route donnée (représentés par les nœuds i et $i+1$), ainsi qu'une autre paire de clients adjacents (représentés par les nœuds j et $j+1$) dans la même route. Si la modification de la combinaison actuelle $(i, i+1)$ en $(i, j+1)$ et $(j, i+1)$ permet de réduire la distance totale du trajet, alors l'opérateur effectue cette modification. Cette opération est répétée jusqu'à ce qu'il n'y ait plus d'opportunité d'optimisation disponible.

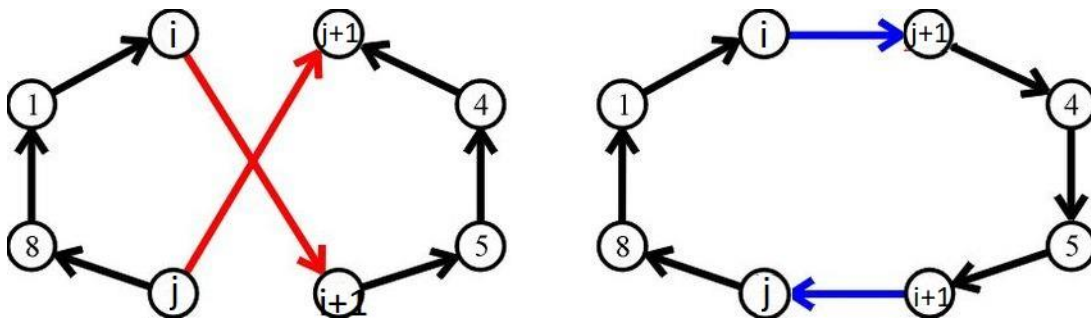


Figure 6 : exemple d'un 2-opt pour une route, source : towardsdatascience.com

Move 1 Point considering Capacity : Contrairement au Move 1 Point, cet opérateur effectue plusieurs recherches de positions en tenant compte des clients situés dans des routes surchargées. Lorsque le poids de la capacité des véhicules dépasse un certain seuil, le nombre de répétitions de l'opérateur augmente pour forcer le respect des contraintes de capacité au cours des itérations.

Exchange 2 Points : Cet opérateur réalise un échange de position entre deux points chaque fois que cette modification conduit à une réduction du coût. Similaire à l'approche du 2-opt, cet opérateur continue de se répéter jusqu'à ce qu'il n'y ait plus d'échanges à effectuer.

4.3.3. Implémentation sur Python

Nous avons choisi d'implémenter cet algorithme en Python pour plusieurs raisons. Tout d'abord, Python offre une grande flexibilité et une syntaxe claire, ce qui permet une mise en œuvre plus rapide et efficace. De plus, Python est aussi avantageux pour la disponibilité de bibliothèques et de modules spécialisés pour l'optimisation, tel que NumPy, Sklearn et Copy, qui facilitent l'implémentation des différentes composantes de l'algorithme.

Pour mettre en œuvre cet algorithme en Python, j'ai développé un programme en utilisant une structure en parties. La première partie contient les positions des nœuds, la deuxième est dédiée à la manipulation du problème de location-routing, et enfin la troisième couche permet de résoudre ce problème. Ces couches sont conçues de manière indépendante, mais elles doivent suivre un seul protocole pour que la sortie d'une couche soit adaptable à l'entrée de la couche en dessous. Ainsi, toute modification apportée à l'une des couches n'aura pas d'impact négatif sur les autres, tant que le protocole est respecté.

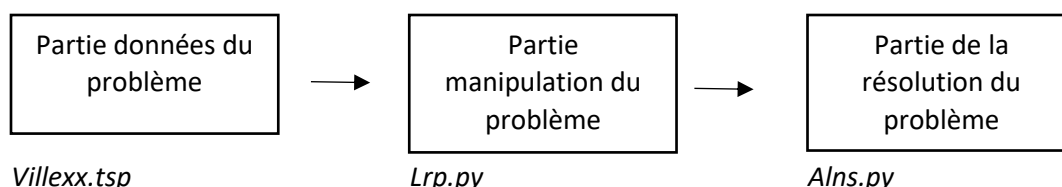


Figure 7 : voir le projet sur https://github.com/ahf0/inrae_stage/blob/main/project/

4.3.3.1. Partie des données

En pratique, cette couche correspond à un fichier texte contenant les positions des nœuds, à la fois des clients et des dépôts. Ce fichier est nommé selon le format suivant : "NomDeLaVille.tsp" sur GitHub. Ce fichier peut également comporter une description générale de l'espace, notamment le nombre de points et leur distribution. Le contenu du fichier doit impérativement inclure l'en-tête "DISPLAY_DATA_SECTION", suivi d'un tableau à trois colonnes comprenant les indices des points, leurs coordonnées en abscisses et en ordonnées, et enfin l'indicateur "EOF" pour marquer la fin du tableau.

```

1  NAME: ville04
2  TYPE: LRP
3  COMMENT: 61 point distribués uniform
4  DIMENSION: 61
5  DISPLAY_DATA_SECTION
6      1      22.0      65.0
7      2      51.0      32.0
8      3       1.0      35.0
9      4      36.0      24.0
10     5      35.0      58.0

```

Figure 8 : exemple de fichier ville04.tsp sur Github

Remarques :

- La rédaction du fichier sur Github convertit les données du texte de str vers bytes, ce qui est nécessaire pour la couche manipulation.
- Alors que les indices débutent à 1, lors de la manipulation, ces indices débutent à 0. Ainsi, l'indice "n" utilisé dans la manipulation correspond à "n+1" dans le fichier.

4.3.3.2. Couche de la manipulation des données

La deuxième couche est une classe Python que nous avons créée. Elle a pour fonction de manipuler les données du fichier "villexx.tsp". Cette classe doit être en mesure d'importer le fichier en utilisant le nom de la ville, de lire les données du fichier, de stocker les positions des points, de calculer les distances entre ces points, de stocker la liste représentant une solution, d'identifier les dépôts et les clients, de stocker les clients et les dépôts non assignés, et enfin de fournir une représentation graphique de cette solution. Ce fichier est nommé "lrp.py" et il contient la classe "LocationRoutingProblem" avec deux méthodes principales :

- `problemLRP.plotData()`² : qui trace un graphique qui représente les points dans leurs positions en rouge, les dépôts en X et les itinéraires par des traits en se basant sur la solution proposée.
- `problemLRP.copy()` : qui copie tous les attributs de l'objet `problemLRP` vers un autre objet.

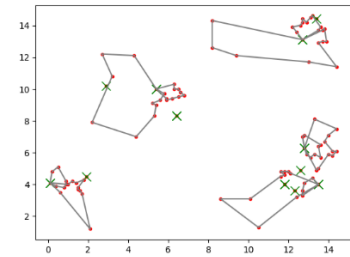


Figure 9 : exemple de la méthode `PlotData`

Chaque objet contient les attributs suivants

```
problemLRP.name
problemLRP.locations
problemLRP.distances
problemLRP.solution # ex : [[depot1, [route11], [route12]], [depot2, [route21], [route22], [route23]]]
problemLRP.lrpSize
problemLRP.client_index
problemLRP.depot_index
problemLRP.unassigned_client
problemLRP.unassigned_depot
```

Remarque :

- Des fonctions à l'intérieur de classes permettent de remplir l'attribut *location* en explorant le fichier *villexx.tsp* et calculer les distances entre les nœuds.
- Nous avons décidé de représenter une solution de la forme suivante :

`problemLRP.solution = [[depot1, [route11], [route12]], [depot2, [route21], [route22], [route23]]]`

La Figure 10 représente un exemple d'un problème LRP avec la solution `[[1, [0,6], [2,3]], [5, [7,11,9]]]`

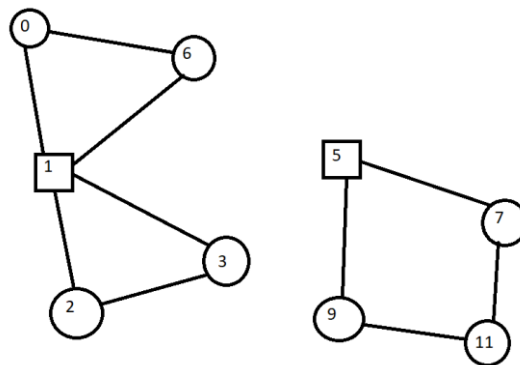


Figure 10 : Représentation graphique de la solution

4.3.3.3. Partie de la résolution

Cette partie représente la section du code qui exploite l'algorithme ALNS pour résoudre le problème de LRP et obtenir une solution pertinente. Elle agit comme une interface permettant de choisir parmi les problèmes spécifiques *villeXX*, définir les fonctions des opérateurs de destruction, réparation et la recherche locale. En outre, il permet de définir les contraintes, les indices des dépôts potentiels, d'établir une solution initiale, de spécifier la condition d'arrêt et d'ajuster les valeurs des paramètres de l'algorithme. De plus, cette couche est en mesure de fournir des informations sur les résultats, telles que la solution finale, le rapport des coûts final et initial, le nombre d'itérations pour atteindre la

² `problemLRP` est ici un objet de classe `LocationRoutingProblem`

dernière réduction, le score final des opérations de destruction et de réparation, ainsi que l'affichage graphique du dernier résultat et des courbes de coûts.

Concrètement, cette couche, appelé *ALNS.py*, représente une traduction en Python de l'algorithme ALNS (consultez la section Algorithme 1), intégrant des lignes de code pour enregistrer, à chaque itération, la valeur des coûts $f(s)$, $f(s')$, $f(s^*)$ ainsi que des poids (*weights*) de la fonction WPN dans des listes, facilitant ainsi leur visualisation graphique.

Finalement, nous avons rajouté une partie pour visualiser la variation de la solution optimale au cours des itérations.

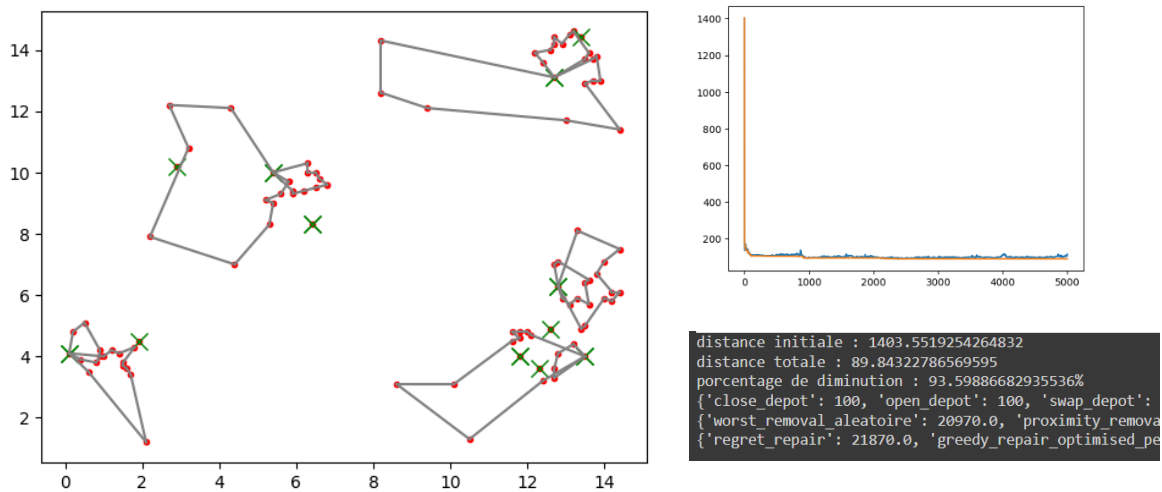


Figure 11: Exemple des résultats d'une manip pour ville09

Il convient de souligner que la Figure 11 représente les résultats d'une expérimentation où nous avons utilisé une solution initiale aléatoire, ce qui explique la valeur extrêmement élevée du coût à l'itération 0.

5. Résultats

Nous présentons ici les résultats obtenus à travers l'application de l'algorithme ALNS au problème de Location-Routing. Ces résultats illustrent l'efficacité de notre approche en termes de performances et de qualité des solutions. Nous examinerons les solutions trouvées pour différents scénarios et configurations, ainsi que les améliorations réalisées grâce aux opérateurs de recherche et aux stratégies d'optimisation utilisés. Il est important de noter que la comparaison de nos résultats avec ceux d'autres études peut s'avérer délicat, étant donné que nos contraintes diffèrent. Pour nos expérimentations, le code a été exécuté sur un processeur Virtual Intel(R) Xeon(R) CPU @ 2.20GHz, en utilisant Google Colab.

5.1. Description des instances

En raison de la nature spécifique de notre problème, nous avons entrepris des tests sur plusieurs instances que nous avons créées afin d'évaluer les performances de notre algorithme dans divers scénarios. Ces tests ont pris en compte différentes variables telles que le nombre de nœuds, la répartition spatiale des points, les paramètres de configuration et les contraintes imposées.

En plus, nous avons basé des instances sur celles utilisées par Tuzun et Burke 1999. Cependant, en raison des différences fondamentales entre notre problème et le leur, nous n'avons répliqué que les positions des dépôts et des clients, ainsi que les coûts d'ouverture des dépôts. Les coûts des véhicules et des routes n'ont pas été pris en compte. De plus, nous avons fixé un nombre maximal de véhicules plutôt que de considérer un coût d'acquisition de véhicules. Nous n'avons pas non plus appliqué la contrainte de capacité des dépôts. Une autre distinction importante réside dans le fait que, pour notre

problème, tous client ont le même nombre de demandes, tandis que Tuzun et Burke ont imposé un nombre de demandes différents pour chaque client.

5.2. Configuration des paramètres

Le choix des valeurs des paramètres revêt une grande importance pour garantir des résultats de qualité. Chaque paramètre exerce une influence sur les solutions générées, et il est complexe de prédire le comportement de l'algorithme en réponse à différentes combinaisons. Bien que nous n'ayons pas élaboré de méthode précise pour déterminer ces valeurs lors de notre stage, nous avons néanmoins identifié des tendances concernant l'impact des paramètres initiaux sur les résultats. Ces observations nous ont grandement éclairés, et voici quelques remarques à ce sujet sur Tableau 1.

Paramètre	Nom	Observations	Valeur choisie
Nombre_iterations	Nombre des itérations : condition d'arrêt	Plus d'itération signifie nécessairement plus de temps d'exécution et plus de solutions explorées.	5000
min_val	Poids minimal	Si le poids minimal est grand, nous explorant les solutions avec des violations juste si ces dernières présentent une réduction importante au niveau de distance ; réduction supérieur au poids.	Moyen des distances entre les nœuds /5
Alpha	θ Dans Algorithme 1	Si grand, ce pourcentage permet d'explorer plus de solution. Si petit, il limite les nombre de solution, et donc moins de temps pour retrouver une solution optimale locale	0.2
Regret_num	Paramètre k du K-regret heuristique de Ropke & Pisinger	Une valeur élevé accroît la qualité des résultats mais augmente fortement les temps de calcul	5
W	W dans Algorithme 1	Une valeur moins élevé augmente le temps de recherche local. Les destruction importante (DL) nécessite une phase plus longue de recherche locale.	120
ScoreDL ScoreDS ScoreR	Listes des Scores au départ	Si un score est plus grand au départ, nous favorisant l'utilisation d'un operateur au début.	Tous égaux
N_min N_max	C'est le nombre de clients détruits par les DS	Une valeur grande signifie un changement important des solution à chaque itération et donc plus de solutions exploré mais moins de optimums locaux.	Entre 6% et 10 % des clients.
Nu	Paramètre dans le mise à jour des scores	Selon l'efficacité de la solution trouvée, Nu, Nu/2 ou Nu/3 est rajouté au score de l'opérateur utilisé.	30
Solution_initiale	Solution initiale	La solution initiale influence les résultats de manière incompréhensible, mais nos expériences ont montré des meilleurs résultats avec des solutions initiales composées par des routes propres pour chaque client.	Route propres pour chaque client.

Tableau 1

5.3. Résultat du modèle

Dans cette section, nous présenterons les résultats de l'exécution de notre modèle python sur plusieurs problèmes, y compris des scénarios que nous avons créés ainsi que des scénarios inspirés d'instances classiques. Nous ferons appel au fichier *alns.py* pour extraire les informations essentielles, telles que le coût à l'itération 50 (étant donné que le coût initial est élevé en raison de la solution initiale), le coût final, la réduction en pourcentage par rapport à l'itération 50, le numéro de l'itération de la dernière solution finale, les variations des poids. Pour toutes les instances, la capacité des véhicules est fixée à 30, et le nombre maximal de véhicules par dépôt est 4.

Nom du problème (P : points D : dépôts)	Coût final	Pourcentage de réduction	Nombre de dépôts ouverts	Nombre de véhicules	Itération de la dernière solution op	Temps d'exécution
Ville03 30p-2d	201.62	17.52%	1	1	1094	1 min
Ville04 60p-2d	480.98	88.25%	1	2	434	4 min
Ville06 100p-5d	89.00	74.87%	3	5	3627	14 min
Ville09 110p-10d	77.42	41.99%	4	5	4841	18 min
Ville02 200p-4d	2867.08	89.13%	2	8	595	55 min

Tableau 2

En analysant Tableau 2, la première observation qui se dégage est la forte augmentation du temps d'exécution en fonction du nombre de nœuds pour le même nombre d'itérations. Cela suggère que notre algorithme intègre des méthodes ou des fonctions de complexité algorithmique exponentielle. Cette augmentation devient particulièrement prononcée lorsque le nombre de points atteint un ordre de grandeur proche de 1000, tel qu'énoncé dans le cahier des charges du projet. Par conséquent, les durées résultantes deviennent extrêmement élevées, au point que les ordinateurs actuels ne peuvent les résoudre dans un délai raisonnable.

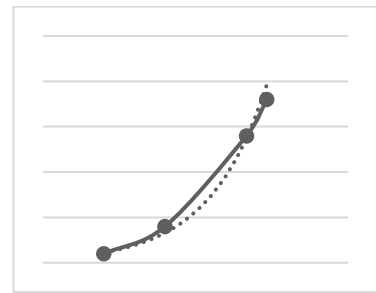


Figure 12 : évolution de temps d'exécution avec le nombre de points

Une autre observation importante que l'on peut tirer des courbes de variation des poids, annexe 1, concerne le comportement du poids associé au nombre de véhicules. Au début des itérations, ce poids tend à s'approcher de sa valeur maximale, pour ensuite diminuer après plusieurs itérations. Cette oscillation est due à la violation de la contrainte liée à ce poids par la solution initiale, qui associe une route distincte à chaque client.

En outre, pour des instances avec un grand nombre de points et des contraintes de capacité et de nombre de véhicules strictes, il est important de noter que l'algorithme ne parvient pas à générer de nombreuses solutions sans violations, nous pourrions visualiser ceci par le graphe de la solution ville04 (voir annexe 1) et à la valeur de la dernière itération de cette ville (voir tableau 2). Ainsi, pour assurer la faisabilité des solutions, il est nécessaire que le produit du nombre de dépôts, du nombre de véhicules et de la capacité de chaque véhicule soit supérieur au nombre total de clients. Cette constatation souligne l'importance de considérer attentivement les contraintes dès le départ de la modélisation et de l'application de l'algorithme.

Les résultats de nos expériences sont en annexe 1.

5.4. Améliorations de l'implémentation du modèle

Les résultats mettent en évidence une tendance où le temps d'exécution augmente de manière exponentielle en fonction du nombre de points. Cette corrélation suggère que l'algorithme utilise des méthodes avec une complexité exponentielle, ce qui pourrait poser des problèmes pour un grand nombre de points.

Pour résoudre cette problématique, deux solutions pourront être étudiées. La première consiste à adapter la recherche locale en poussant l'exploration au-delà de la première amélioration, permettant ainsi une exploration plus approfondie. La seconde solution implique une optimisation des calculs des positions possibles des clients, en se concentrant spécifiquement sur les routes à proximité de chaque client ou explorées les possibilités de lier ce client aux clients proche, **cela nécessite de stocker les informations des emplacements des clients**, ce qui va nécessairement rendre la complexité linéaire.

Nous pourrions également envisager l'utilisation d'autres langages de programmation pour la rédaction de notre code, comme C++. Ce dernier pourrait potentiellement être jusqu'à 100 fois plus rapide que Python.

Voir le projet, code alns.py, code expliqué sur notebook, classe lrp.py et animation en GIF sur Github:

<https://l.linklyhq.com/l/1tFxD>

6. Bilan du stage

Mon stage a été une expérience enrichissante et passionnante. Il m'a permis de plonger dans le monde de la recherche et de découvrir le rôle d'un ingénieur de recherche. En particulier, j'ai pu explorer le domaine de l'optimisation combinatoire, qui revêt une importance cruciale dans la résolution de problèmes complexes tels que la gestion des déchets.

J'ai apprécié l'opportunité d'appliquer les concepts théoriques que j'ai appris dans un contexte pratique. Travailler sur le problème de Location-Routing m'a permis de mettre en pratique mes compétences en programmation et en modélisation, tout en comprenant les enjeux réels liés à la gestion des déchets et à la logistique.

Durant ce stage, j'ai saisi l'importance fondamentale de la compréhension approfondie du sujet et de la mise en place d'un plan de travail bien structuré. Cette prise de conscience s'est manifestée alors que je faisais face à des défis techniques, en particulier lorsque j'ai découvert que nos contraintes différaient de celles des autres problèmes similaires. Cette divergence a engendré des difficultés lors de la comparaison des résultats.

J'ai également réalisé l'importance de la communication au sein d'une équipe de recherche. Si j'avais consulté mes encadrants plus régulièrement, j'aurais probablement évité de perdre du temps sur certaines difficultés techniques. Cette expérience m'a clairement montré que la communication ouverte avec les membres expérimentés de l'équipe peut accélérer la résolution des problèmes et faciliter la progression vers les objectifs du projet.

En résumé, ce stage m'a permis de découvrir concrètement le rôle de l'ingénieur de recherche, en mettant en lumière les domaines clés tels que l'optimisation combinatoire et la gestion des déchets. Cette expérience a renforcé mon désir de continuer à enrichir mes compétences et m'a motivé à contribuer aux progrès techniques visant à une meilleure gestion des transports, un défi crucial pour faire face aux enjeux énergétiques et climatiques actuels.

6.1. Réflexions sur les avancées

Malgré l'avancement remarquable que j'ai fait dans ce projet, je n'ai pas pu accomplir plusieurs tâches importantes pour des raisons de la durée de stage et le manque de l'expérience dans la programmation. Parmi ces tâches essayer le programme dans une territoire cible, et la traduction de l'algorithme sur Julia ou C++.

Autre tâches non faite, c'est résoudre le problème de la complexité de l'algorithme, une des solutions proposé est qui semble aussi prometteuse, c'est après la suppression des clients, les opérateurs de

réparation peuvent évaluer l'insertion de clients supprimés dans des positions limitées. Comme les routes proches ou les insérer à côté des clients à proximité, cela nécessite d'utiliser un variable pour stocker les positions des clients dans la solution.

7. Conclusion

En conclusion, ce stage a été une expérience exceptionnelle qui m'a permis de plonger profondément dans le domaine complexe de la recherche en optimisation combinatoire appliquée à la gestion des déchets et au problème de Location-Routing. Au cours de ces mois passés au sein de l'équipe de recherche, j'ai acquis des connaissances et des compétences qui ont largement dépassé mes attentes initiales.

J'ai pu constater à quel point la résolution du problème de Location-Routing est cruciale dans le contexte de la gestion efficace des ressources de transport et de la minimisation des coûts associés. L'algorithme ALNS s'est révélé être un outil puissant pour aborder ces problématiques complexes et pour obtenir des solutions pratiques et efficaces. Les différentes couches de l'algorithme, de l'initialisation à la recherche locale, ont été un terrain d'apprentissage passionnant et exigeant.

Ce stage m'a également fait prendre conscience de l'importance de la méthodologie de recherche rigoureuse. J'ai appris que la planification minutieuse, l'analyse des résultats intermédiaires et l'adaptation des paramètres jouent un rôle crucial dans l'obtention de solutions de haute qualité. L'ajustement dynamique des poids dans la fonction de pénalité a été particulièrement instructif, montrant comment une approche intelligente peut favoriser l'exploration efficace de l'espace des solutions.

En collaborant avec mes encadrants et en échangeant avec mes collègues, j'ai pris conscience de l'importance de la communication dans un environnement de recherche. La capacité de solliciter des opinions, de partager des idées et de résoudre ensemble les défis techniques a été une leçon précieuse qui m'a montré à quel point une équipe interdisciplinaire peut être productive.

Je tiens également à souligner les défis que j'ai rencontrés, notamment la complexité croissante des instances de test et la nécessité d'adapter les stratégies de recherche pour gérer cette complexité. J'ai compris que la recherche d'une solution optimale est un équilibre subtil entre les performances computationnelles et la qualité des solutions obtenues.

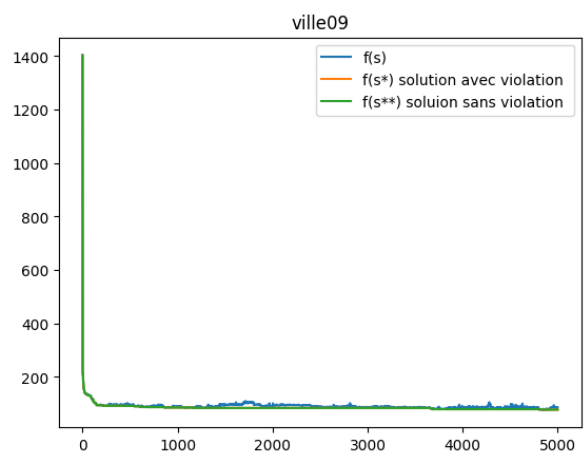
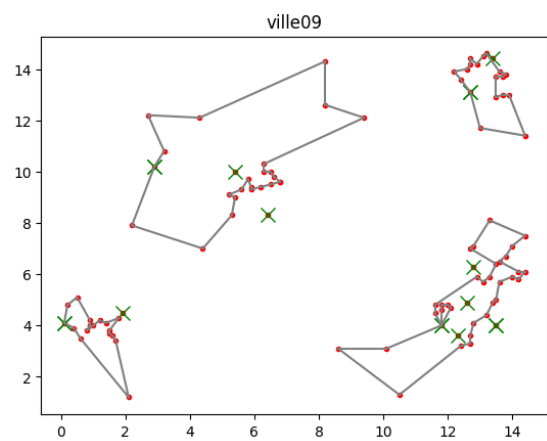
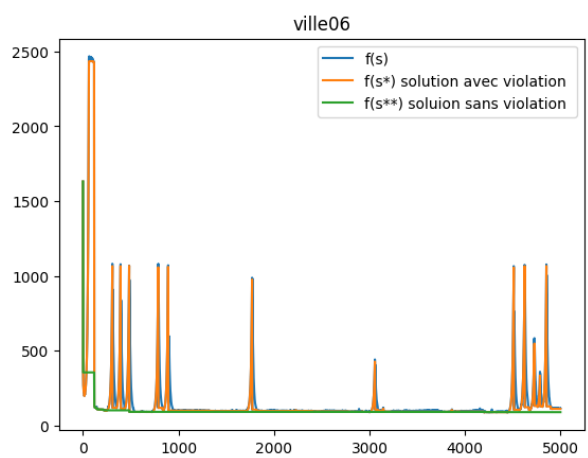
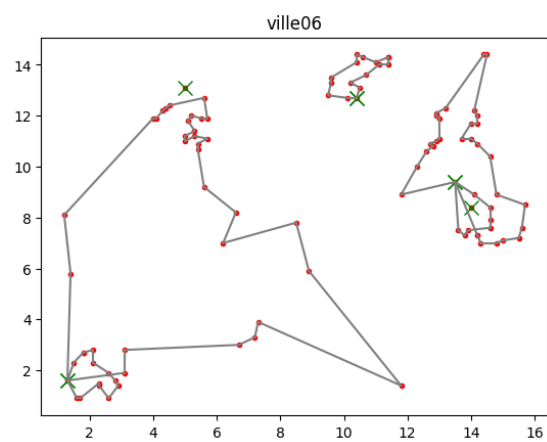
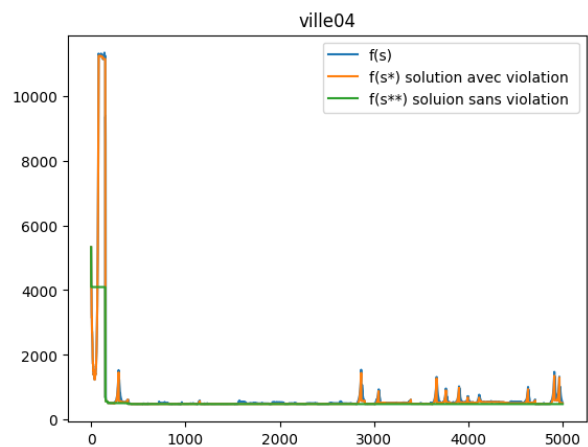
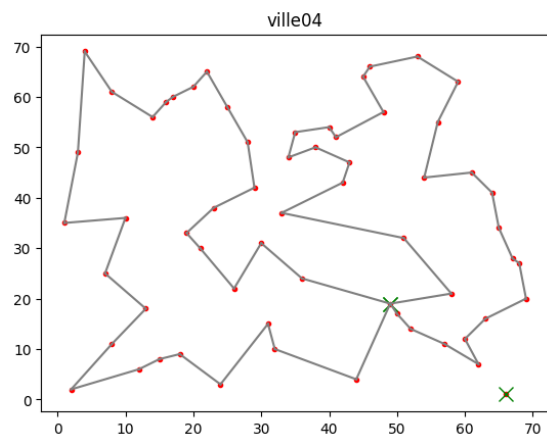
En fin de compte, ce stage a solidifié mon intérêt pour la recherche en optimisation combinatoire et son application concrète aux problèmes environnementaux. Il m'a également encouragé à continuer à développer mes compétences techniques et mon expertise dans ce domaine. Je suis convaincu que les compétences et les connaissances que j'ai acquises au cours de ce stage me serviront dans ma future carrière professionnelle, que ce soit dans la recherche académique ou dans l'industrie, et me permettront de contribuer de manière significative à la résolution des défis complexes auxquels notre société est confrontée en matière de durabilité et d'efficacité.

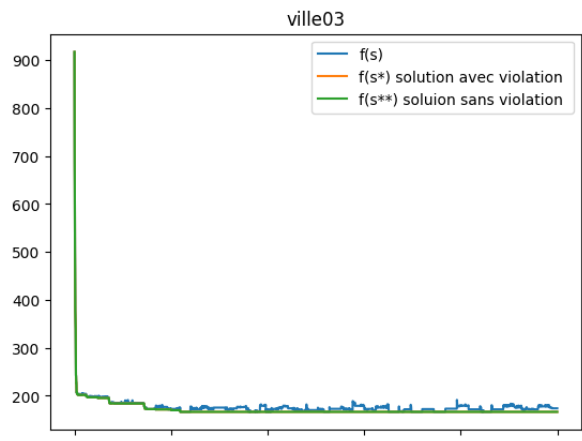
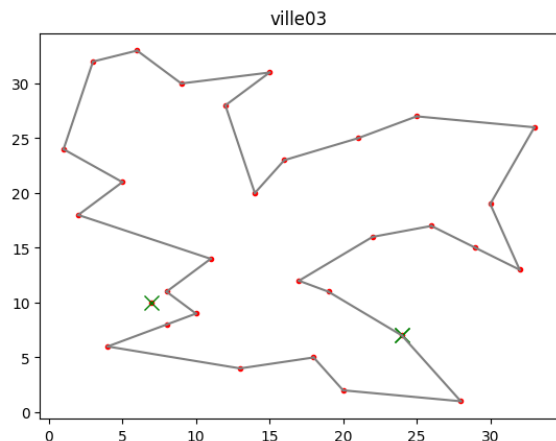
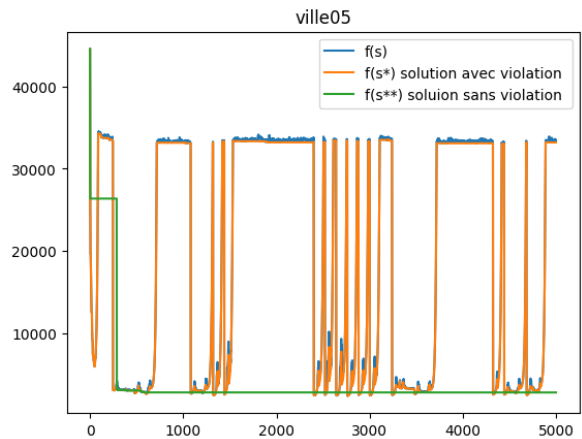
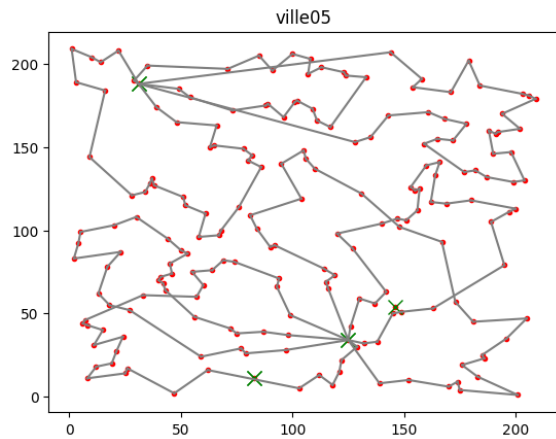
8. Bibliographie

- [1] « Biodéchets », *Ministères Écologie Énergie Territoires*. <https://www.ecologie.gouv.fr/biodechets> (consulté le 16 août 2023).
- [2] Y. Marinakis, « Location routing problemLocation Routing Problem », in *Encyclopedia of Optimization*, C. A. Floudas et P. M. Pardalos, Éd., Boston, MA: Springer US, 2009, p. 1919-1925. doi: 10.1007/978-0-387-74759-0_345.
- [3] « Prévenir les risques dans la filière de valorisation des biodéchets - Actualité - INRS ». <https://www.inrs.fr/actualites/prevenir-risques-filiere-valorisation-biodechets.html> (consulté le 16 août 2023).
- [4] « Déchets chiffres-clés : L'essentiel 2021 », *La librairie ADEME*. <https://librairie.ademe.fr/dechets-economie-circulaire/5417-dechets-chiffres-cles-l-essentiel-2021-9791029719622.html> (consulté le 16 août 2023).
- [5] S. T. W. Mara, R. J. Kuo, et A. M. S. Asih, « Location-routing problem: a classification of recent research », *Intl. Trans. in Op. Res.*, vol. 28, n° 6, p. 2941-2983, nov. 2021, doi: 10.1111/itor.12950.
- [6] S. Ropke et D. Pisinger, « An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows », *Transportation Science*, vol. 40, p. 455-472, nov. 2006, doi: 10.1287/trsc.1050.0135.
- [7] L. Perron, P. Shaw, et I. Sa, « Parallel Large Neighborhood Search », oct. 2003.
- [8] R. Lutz, « Adaptive Large Neighborhood Search », Universität Ulm. Fakultät für Ingenieurwissenschaften und Informatik, Ulm, 2015.
- [9] V. C. Hemmelmayr, J.-F. Cordeau, et T. G. Crainic, « An adaptive large neighborhood search heuristic for Two-Echelon Vehicle Routing Problems arising in city logistics », *Computers & Operations Research*, vol. 39, n° 12, p. 3215-3228, déc. 2012, doi: 10.1016/j.cor.2012.04.007.
- [10] E. H. L. Aarts et J. K. Lenstra, *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

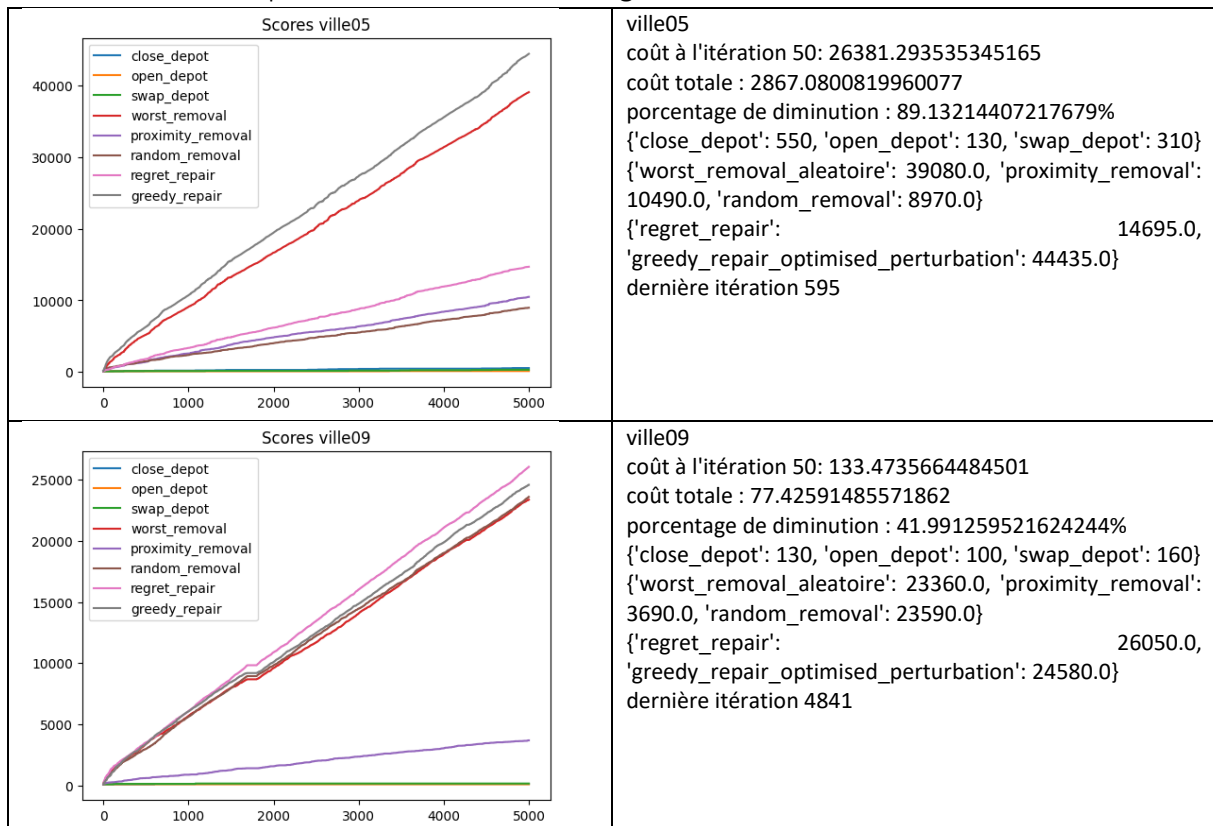
9. Annexe 1

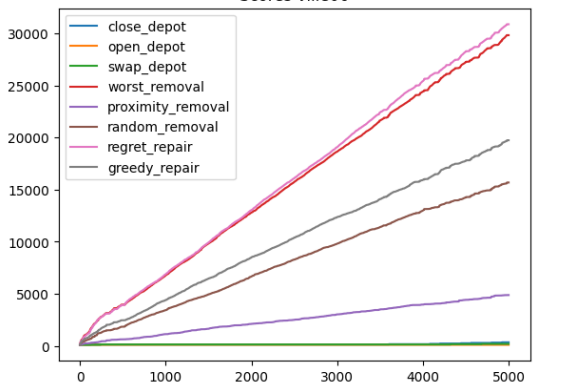
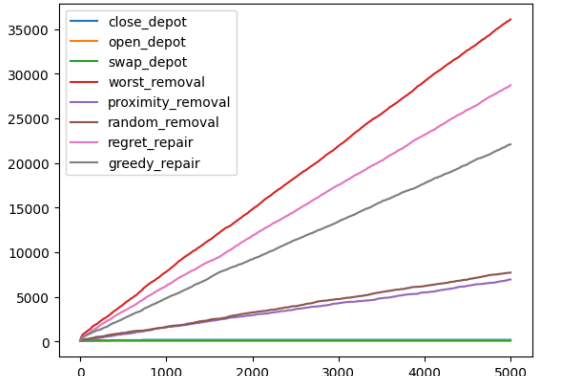
9.1. Graphe représentatif des solutions et variation des coûts





9.2. Scores des opérateurs et informations générales



<p>Scores ville06</p> 	<p>ville06</p> <p>coût à l'itération 50: 354.2807422348</p> <p>coût totale : 89.00203904184252</p> <p>pourcentage de diminution : 74.87789280914855%</p> <p>{'close_depot': 340, 'open_depot': 100, 'swap_depot': 190}</p> <p>{'worst_removal_aleatoire': 29820.0, 'proximity_removal': 4875.0, 'random_removal': 15680.0}</p> <p>{'regret_repair': 30870.0, 'greedy_repair_optimised_perturbation': 19735.0}</p> <p>dernière itération 3627</p>
<p>Scores ville03</p> 	<p>ville03</p> <p>coût à l'itération 50: 201.62696072834464</p> <p>coût totale : 166.29780869373545</p> <p>pourcentage de diminution : 17.521950778547282%</p> <p>{'close_depot': 190, 'open_depot': 130, 'swap_depot': 100}</p> <p>{'worst_removal_aleatoire': 36100.0, 'proximity_removal': 6960.0, 'random_removal': 7730.0}</p> <p>{'regret_repair': 28720.0, 'greedy_repair_optimised_perturbation': 22090.0}</p> <p>dernière itération 1094</p>