Rapport de projet tutoré de 5ème année

Sujet : Recherche opérationnelle pour l'optimisation des flux de brancardiers à l'hôpital

Camille Carbasse, Quentin Diaz, Amine El Harchaoui

Client : Dimitri DUVAL

Tuteur école : Hervé PINGAUD

Remerciements

Nous souhaitions tout d'abord remercier vivement M.PINGAUD pour toute l'aide qu'il nous a apportée sur ce projet, pour le temps qu'il nous a consacré, pour son expertise et ses conseils.

Nous adressons également nos remerciements à l'entreprise Atout Majeur Concept et en particulier à M.DUVAL sans qui nous n'aurions pu travailler sur ce projet. Nous remercions également Damien DE QUINA et Victor MAINTENANT qui, avec M.DUVAL, nous ont fourni tous les documents nécessaires pour mener à bien ce projet et se sont rendus disponibles tout du long.

Enfin, nous aimerions remercier Geoffrey DE SMET pour son aide précieuse sur les forums dédiés à OptaPlanner.

Résumé

Dans le cadre de notre projet tutoré de 5ème année à ISIS, nous avons réalisé le projet proposé par Atout Majeur Concept, qui porte sur la recherche opérationnelle visant à optimiser les flux de brancardiers à l'hôpital. Ce projet est divisé en deux étapes clés. Tout d'abord, nous avons étudié les solutions existantes chez AMC, à savoir la méthode heuristique actuellement utilisée par l'entreprise, ainsi que la méthode exacte qui a été mise en œuvre lors d'un stage l'année dernière par deux étudiants d'ISIS. En parallèle, nous avons développé une solution métaheuristique. La deuxième étape consiste à améliorer la méthode exacte et à comparer les performances (rapidité, précision) des méthodes heuristique, métaheuristique et exacte. Cette comparaison vise à permettre à AMC de développer sa culture de la recherche opérationnelle et de soutenir sa R&D afin de proposer une solution à ses clientes.

Abstract

As part of our fifth-year supervised project at ISIS, we carried out the project proposed by Atout Majeur Concept, which focuses on operational research aimed at optimizing the flow of orderlies in the hospital. This project is divided into two key stages. Firstly, we studied the existing solutions at AMC, namely the heuristic method currently used by the company, as well as the exact method that was implemented during an internship last year by two ISIS students. In parallel, we developed a metaheuristic solution. The second stage involves improving the exact method and comparing the performance (speed, accuracy) of the heuristic, metaheuristic, and exact methods. This comparison aims to enable AMC to develop its culture of operational research and support its R&D efforts in order to provide a solution to its clients.

Table des matières

I. Introduction

1. Contexte

Depuis près de 20 ans, Atout Majeur Concept s'engage aux côtés des établissements de santé et des GHT (Groupements Hospitaliers de Territoire), en proposant des solutions digitales visant à améliorer leur gestion globale et renforcer leur communication. Leur objectif principal est de dématérialiser les processus et de favoriser la transformation digitale des établissements.

Atout Majeur Concept a développé plusieurs solutions, dont le Progiciel de Gestion Intégré SESAME, un outil puissant qui intègre plus de 65 applications répondant aux différentes problématiques organisationnelles, stratégiques, administratives et de diffusion de l'information. Le positionnement de ce progiciel ainsi que la gestion des flux patients en font son succès.

Grâce à leur riche patrimoine de données et à leurs relations étroites avec des établissements hospitaliers innovants, Atout Majeur Concept soutient activement la recherche en optimisation de la gestion des ressources modélisée par ce flux de patients. Ils se concentrent entre autres sur les flux des brancardiers au sein des établissements de santé.

2. Enjeux et Objectif

Le projet "Recherche opérationnelle d'optimisation pour les flux de brancardiers à l'hôpital" a pour objectif de comparer différentes approches, méthodes et outils de recherche opérationnelle en matière d'optimisation. La comparaison se concentre principalement sur trois méthodes : heuristique, métaheuristique et exacte, pour l'optimisation des flux de brancardiers.

Ce projet vise à poursuivre le développement des outils de gestion optimisée des tournées de brancardiers, initié précédemment par des étudiants de l'école d'ingénieurs ISIS en relation avec un contrat de recherche. Le travail consistera à évaluer les limites des modèles développés par ces travaux par rapport aux besoins actuels. Une autre facette du projet consistera à présenter de manière pédagogique les capacités de résolution de trois approches méthodologiques en recherche opérationnelle : l'approche heuristique avec le programme fourni par AMC, l'approche méta heuristique avec l'outil OptaPlanner, et l'approche exacte basée sur la programmation linéaire.

Cette comparaison permettra à AMC de choisir la méthode qui offre les meilleurs résultats dans un délai minimal (inférieur à une dizaine de minutes). L'optimisation choisie devra minimiser une fonction objectif qui prend en compte le temps d'attente pour le patient.

En comparant les trois méthodes en supposant que les modèles sont équivalents dans leur implémentation respective, cela permettra soit de confirmer l'utilisation de la méthode heuristique et d'évaluer la pertinence des règles et de l'algorithme utilisés, soit d'envisager un changement de stratégie en poussant la recherche et le développement pour proposer une solution satisfaisante. Dans le premier cas, il s'agira de prouver que la méthode heuristique est la meilleure en termes de temps d'exécution et de fiabilité de l'optimisation, tandis que dans le second cas, il s'agira de démontrer qu'une des deux autres méthodes surpasse la méthode actuellement en place en termes de bénéfices en temps et/ou en optimisation. Nous avons travaillé sur un jeu de données fourni par AMC. Ce jeu de données provient d'une journée réelle vécue par une équipe de brancardiers au sein d'un hôpital. L'étude de ce jeu de données réelles nous a permis de travailler sur un cas concret qui sera reproduit par la suite.

II. Présentation du sujet

1. Brancardage

Il est essentiel de comprendre ce qu'est le brancardage et les enjeux qui y sont associés, étant donné que notre sujet porte sur les flux de brancardiers. Le brancardage joue un rôle crucial dans la gestion des flux de patients et dans la prévention de la congestion au sein des établissements de santé.

Le brancardage englobe différents types de transports :

- Les déplacements internes au sein du service (transferts entre les boxes et les salles d'attente),
- Les transports liés aux plateaux médico-techniques (salles d'opération, imagerie, etc.)
- Les transports vers les services cliniques d'hospitalisation.

Il s'agit d'un domaine majeur de la prise en charge des patients dans les établissements de santé, car il a un impact direct sur les soins qui leur sont prodigués. Le processus de brancardage comprend plusieurs éléments :

- Le transport des lits, des brancards et d'autres équipements vers les patients appropriés.
- La prise en charge rapide des patients sans les brusquer afin d'éviter tout accident.
- Le transport précautionneux des patients vers les services appropriés.
- Le dépôt des patients.

Du fait de sa transversalité, le brancardage est souvent traité séparément des services. Il peut être exclu des études sur le fonctionnement des services, alors qu'il est le moyen d'entrée et de sortie de ceux-ci. Il est donc crucial de ne pas négliger la gestion et l'optimisation des flux de brancardage, car cela peut avoir un impact significatif et positif. En réduisant les retards et les erreurs, la qualité des soins aux patients s'en trouve indirectement améliorée.

L'expression "flux de brancardiers" décrit en réalité les tournées effectuées par les brancardiers au sein d'un ou de plusieurs établissements. Dans le cadre de notre projet, nous avons travaillé avec des jeux de données modélisant ces tournées, comprenant le nombre de brancardiers par jour, leurs horaires de début, de fin de journée et de pause déjeuner, les durées et les horaires de début de chaque mission, ainsi que les temps de déplacement d'une mission à une autre. Un flux de brancardiers typique représente tous les déplacements effectués par chaque brancardier au cours de sa journée de travail pour accomplir les missions qui lui sont assignées.

2. Vehicle Routing Problem[1]

Le VRP (Vehicle Routing Problem), ou problème de tournées de véhicules en français, est un problème classique d'optimisation de la logistique et de la planification des déplacements. Il s'agit de déterminer les itinéraires optimaux pour une flotte de véhicules afin de desservir un ensemble de clients à partir d'un dépôt tout en respectant certaines contraintes.

Le VRP trouve des applications dans de nombreux domaines, tels que la distribution de marchandises, la collecte de déchets, les services de maintenance et de réparation, les

[1] Bouabdallah, M.N., Rached, M., Fondrevelle, J., Bahroun, Z., (2013). “Organization and management of hospital patient transportation system”.

livraisons de colis, et bien d'autres. Son objectif principal est de minimiser les coûts de transport, le temps de trajet ou la distance parcourue, tout en respectant des contraintes spécifiques, telles que la capacité des véhicules, les fenêtres horaires des clients ou les priorités de livraison.

La résolution du VRP est un défi complexe, car il nécessite de trouver un compromis entre l'optimisation des itinéraires et la satisfaction des contraintes. Il existe plusieurs variantes du VRP, notamment le VRP avec fenêtres horaires, le VRP avec capacité, le VRP avec contraintes de temps, le VRP multi-dépôts, etc. A chaque variantes ajoutent des contraintes supplémentaires au problème de base. Toutes ces formes possèdent des différences et ne sont donc pas utilisables dans toutes les situations.

Acronymes	Modèles	Différence avec le VRP
CVRP	Capacitated vehicle routing problem	Ajout d'une capacité associée à une charge de produits transportés par un véhiculee
VRPTW	Vehicle Routing Problem with Time Windows	Ajout d'une fenêtre temporelle delivraison
PVRP	Period Vehicle Routing Problem	Ajout d'une fréquence de visite
IRP	Inventory Routing Problem	Ajout d'une problématique de stock
M-VRPTW	M-Vehicle Routing Problem with Time Windows	Ajout d'une limite du nombre de véhicules

Tableau 1 : Présentation des différents modèles

Pour résoudre le VRP, différentes méthodes sont utilisées, telles que les heuristiques, les métaheuristiques (comme la recherche taboue, les algorithmes génétiques, le recuit simulé) et les méthodes exactes (comme la programmation linéaire, la programmation par contraintes). Les approches utilisées dépendent de la taille de l'instance du problème et de la précision requise dans la solution. La résolution efficace du VRP permet d'optimiser les opérations logistiques, d'améliorer l'utilisation des ressources, de réduire les coûts de transport et d'optimiser les délais de livraison. Cela peut conduire à des économies significatives et à une meilleure satisfaction des clients.

En résumé, le VRP est un problème d'optimisation complexe qui vise à déterminer les meilleures tournées au sens d'un critère de performance choisi pour un ensemble de véhicules afin de servir des clients, tout en respectant des contraintes opérationnelles de qualité de service. Sa résolution permet d'améliorer l'efficacité des opérations logistiques et de réduire les coûts de transport.

3. Méthodes d'optimisation[2]

L'optimisation combinatoire est une branche de l'optimisation mathématique qui traite des problèmes de recherche de la meilleure solution parmi un ensemble fini de possibilités. Ces

[2] Dr. Kherici N. (2020). "Méthode de résolution en optimisation combinatoire".

problèmes impliquent souvent des choix discrets et des combinaisons d'objets ou de variables, d'où le terme "combinatoire".

Les problèmes d'optimisation combinatoire sont présents dans de nombreux domaines, tels que la logistique, la planification, les réseaux, la finance, l'informatique, la biologie, etc. Ils se caractérisent par le fait que la taille de l'ensemble des solutions possibles est généralement très grande, voire exponentielle, rendant la recherche de la solution optimale difficile, voire impossible dans certains cas.

L'objectif de l'optimisation combinatoire est de trouver une solution optimale (ou proche de l'optimal) en explorant efficacement l'espace des solutions. Pour cela, différentes méthodes sont utilisées, notamment :

1. **Les méthodes exactes** : Ces méthodes garantissent de trouver la solution optimale, si elle existe, en examinant toutes les possibilités. Elles comprennent la programmation linéaire, la programmation par contraintes, la programmation dynamique, et l'utilisation de modèles mathématiques spécifiques pour résoudre des problèmes particuliers. Elles sont chronophages car elles parcourent tout l'espace de recherche.

2. **Les heuristiques :** Les heuristiques sont des approches de recherche basées sur des règles empiriques ou des stratégies de recherche guidées. Elles cherchent à fournir des solutions de bonne qualité, mais sans garantie d'optimalité. Les heuristiques couramment utilisées comprennent la recherche locale pour l'utilisation de règles métier.

3. **Les métaheuristiques** : Les métaheuristiques sont des méthodes générales de résolution de problèmes d'optimisation combinatoire. Elles sont souvent basées sur des principes d'exploration et d'exploitation pour rechercher des solutions de qualité sans explorer l'espace de manière exhaustive, mais astucieusement. Les métaheuristiques populaires comprennent la recherche en essaim, les algorithmes évolutionnaires, le recuit simulé, la recherche tabou, et la recherche de voisins.

L'optimisation combinatoire présente des défis importants en raison de la complexité des problèmes et de la taille de l'espace des solutions avec plusieurs milliers de contraintes et de variables de décision. La recherche de solutions efficaces nécessite souvent une combinaison de différentes méthodes, adaptées au problème spécifique et aux contraintes de temps. De plus, l'utilisation de techniques d'optimisation combinatoire peut être enrichie par des approches de modélisation mathématique, des techniques d'apprentissage automatique, et des méthodes d'optimisation hybrides.

III. Principe et hypothèses

Nous avons étudié trois méthodes de résolution pour le Problème de Tournées de Véhicules (VRP) afin de déterminer celle qui répond le mieux aux besoins de notre client. Malgré les différences entre ces méthodes, le problème reste essentiellement le même, avec des principes et des hypothèses communs.

Voici comment nous avons défini le problème et les règles de résolution pour pouvoir les comparer :

1. Chaque mission est caractérisée par une heure de départ et une durée de trajet, qui inclut la prise en charge du patient et le transport entre les services.

2. Il existe deux types de missions : celles qui nécessitent un seul brancardier et celles qui nécessitent deux brancardiers.

3. Chaque brancardier a une durée maximale de travail quotidienne à respecter.

4. Chaque brancardier ne peut être affecté qu'aux missions qui se situent dans ses horaires de travail quotidiens.

5. Il est important de répartir équitablement les missions entre les brancardiers, de manière à ce qu'ils aient un temps de travail ou un nombre de missions similaires.

6. Les deux brancardiers affectés à une même mission doivent obligatoirement partir ensemble.

7. Un brancardier ne peut pas commencer une mission tant qu'il n'a pas terminé une mission précédente qui lui a été attribuée. Il doit y avoir une continuité temporelle entre les missions des brancardiers.

8. Il est nécessaire de minimiser le temps de trajet à vide entre deux missions pour chaque brancardier.

Une fois les règles assimilées, nous avons repris un schéma illustrant la composition de chaque mission (Figure 1) réalisé par le groupe d'étudiant précédent ainsi que le déroulement de certaines missions impliquant plusieurs brancardiers (Figure 2) également réalisé par le groupe d'étudiant. Ces visualisations nous ont permis de mieux appréhender les objectifs de notre étude.

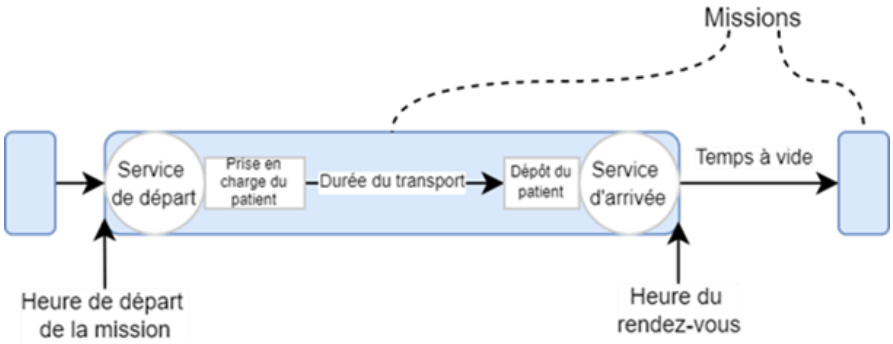

Figure 1: Schéma de composition des missions

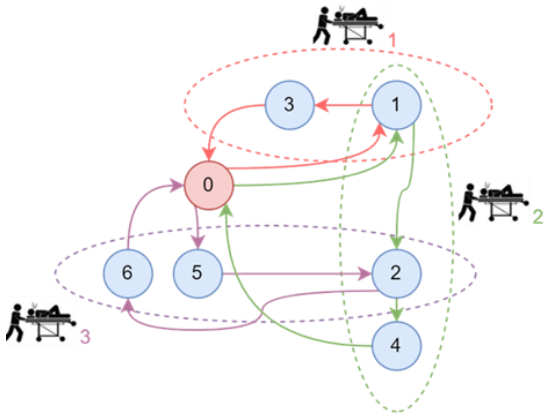

Figure 2: Exemple d'enchaînement des missions

Ainsi, si nous prenons l'exemple de la mission 1 de la Figure 2 le brancardier 1 et le brancardier 2 doivent arriver ensemble au service de départ pour pouvoir suivre la mission comme décrite dans la Figure 1. Ils doivent se synchroniser afin qu'ils puissent prendre en charge le patient.

IV. Méthode heuristique

Les heuristiques sont des méthodes d'optimisation approximatives qui ne garantissent pas la solution optimale d'un problème, mais qui fournissent néanmoins une solution de bonne qualité. Une heuristique est un algorithme qui permet de trouver une solution réalisable dans un délai raisonnable, en se basant sur une fonction objectif. Ce type de méthode reflète une approche similaire à celle que nous adopterons manuellement pour résoudre le problème à partir d'un jeu de règles et d'un algorithme rythmant l'influence des règles .

Dans le cadre de nos comparaisons avec d'autres méthodes de résolution, nous avons utilisé l'heuristique existante du logiciel d'AMC. Cette méthode est implémentée en Python et traite des données provenant de plusieurs fichiers Excel contenant les informations sur les différentes missions d'une journée, les disponibilités des différents brancardiers et une matrice des temps de trajet entre les différents services de l'établissement de santé.

Nous avons principalement examiné le code et la documentation de cette méthode afin de vérifier qu'elle respecte les principes énoncés précédemment et de comprendre son fonctionnement.

L'algorithme se divise en deux grandes parties : la récupération des données et l'assignation de valeurs pour les missions, puis l'affectation des brancardiers optimaux aux missions. Le processus global se rapproche des étapes suivantes :

1. Traitement des missions

   1. **Récupération des données** : les informations sur les missions, les brancardiers disponibles et les temps de trajet entre les services sont extraites des fichiers Excel.

   2. **Assignation de valeurs pour les missions** : Affectations des temps de trajets pour les missions en fonction des deux services de la mission et de la matrice de temps, ainsi que d'une priorité aux missions en fonction là aussi des services, car ils possèdent tous un ordre de priorité

   3. **Affectation des brancardiers aux missions** : un processus itératif est utilisé pour sélectionner les brancardiers "optimaux" pour chaque mission. Différents critères peuvent être pris en compte, tels que la disponibilité des brancardiers, l'équilibre de charge de travail et la minimisation du temps de trajet à vide.

2. Affectation des brancardiers

   - Recherche des brancardiers disponibles pour une mission en tenant compte de l'heure de départ, de leurs horaires de travail et de leur disponibilité actuelle. Si aucun brancardier n'est disponible à ce moment-là, on introduit un retard dans la mission et on recommence la recherche de brancardiers disponibles.
   - Attribution d'un score à chaque brancardier pour déterminer celui qui est le mieux qualifié pour effectuer la mission. Le score est calculé en fonction de la charge de travail déjà effectuée par le brancardier au cours de la journée, de sa proximité avec la mission et de sa capacité à prendre en charge le transport dans les délais impartis.
   - Tri des brancardiers en fonction de leur score et sélection du brancardier ou des brancardiers ayant le meilleur score pour effectuer le transport. Si aucun brancardier n'est disponible, les brancardiers potentiellement disponibles pour le prochain trajet sont exclus de la liste, et les premiers brancardiers restants sont attribués à la mission.

- Une fois que tous les trajets ont été attribués, les résultats sont extraits dans un fichier Excel.

Ces étapes permettent une affectation des brancardiers aux missions en prenant en compte leur disponibilité, leur charge de travail, leur proximité et leur capacité à réaliser le transport dans les délais. Le processus itératif de recherche des brancardiers disponibles et d'attribution des scores garantit que les missions sont assignées de manière efficace et équilibrée, en maximisant la satisfaction des contraintes.

À la fin du processus, les résultats obtenus sont enregistrés dans un fichier Excel, ce qui permet une analyse ultérieure et une évaluation des performances du système d'affectation des brancardiers.

En utilisant cette heuristique, nous obtenons une solution réalisable en un temps raisonnable, tout en prenant en compte les contraintes et en cherchant à optimiser la fonction objectif choisie qui est la minimisation des retards.

La méthode heuristique fournie par AMC, en raison de son fonctionnement, ne nécessite pas de modifications pour être comparée à nos autres modèles, car elle est conforme aux principes énoncés précédemment.

V. Méthode exacte : Programmation linéaire

1. Modèle mathématique

Le modèle mathématique est disponible en annexe.

1) Présentation de l'existant

Les méthodes exactes permettent de donner la solution la plus optimale parmi les nombreuses solutions qu'on peut trouver à un problème posé. Pour le travail que nous avons fourni dans ce projet, nous nous sommes concentrés sur la méthode exacte par programmation linéaire en variable mixte.

La programmation linéaire est une technique mathématique utilisée pour résoudre des problèmes d'optimisation. Elle s'applique aux situations où l'on cherche à maximiser ou minimiser une fonction linéaire, appelée fonction objectif, soumise à un ensemble de contraintes linéaires. Ces contraintes définissent un modèle de problème à résoudre, incluant les limites et les conditions auxquelles la solution doit être conforme.

Le modèle de programmation linéaire repose sur des variables de décision qui représentent les choix à déterminer. La fonction objectif est une combinaison linéaire de ces variables, et les contraintes sont des inégalités ou des égalités linéaires qui définissent les relations entre les variables.

L'objectif de la programmation linéaire est de trouver les valeurs des variables de décision qui optimisent la fonction objectif tout en respectant les contraintes. Cela implique la recherche du meilleur compromis possible dans le champ des contraintes.

Nous n'avons pas commencé à partir d'une page blanche pour développer le modèle linéaire mathématique. Nous avons consulté la documentation d'AMC sur le sujet, en particulier le travail réalisé en stage par Nelson ROGERS et les résultats du PTUT des étudiants qui ont précédemment travaillé sur ce projet. Nelson avait proposé une approche basée sur le VRPTW (Vehicle Routing Problem with Time Windows), qui permet d'introduire des fenêtres de temps flexibles pour les opérations. Cela signifie que le temps d'action n'est plus fixe, mais peut être effectué dans une plage horaire donnée avec un retard toléré.

En examinant le travail précédent, notre groupe a identifié certaines incohérences que nous avons corrigées afin de proposer un modèle mathématique plus précis et cohérent.

2) Améliorations apportées

a) Paramètres et variables

L'uniformisation des variables est une étape importante dans la formulation de problèmes de programmation linéaire. En programmation linéaire, les variables peuvent avoir des unités et des échelles différentes, ce qui peut rendre la comparaison et l'analyse des résultats difficiles.

Dans le cadre de nos modifications, nous avons pris la décision de faire des ajustements aux paramètres et variables existants. Par exemple, le paramètre $Trdv_i$ a été renommé $Hrdv_i$ et la variable $Tdep_{ik}$a été renommée $Hdep_{ik}$. Cette modification a été entreprise pour éliminer toute confusion, car la lettre T pouvait prêter à confusion en suggérant une notion de durée alors qu'il s'agit en réalité d'heures. Cette adaptation a permis de dissiper toute incompréhension lors de l'utilisation de ces paramètres et variables.

Une autre modification a été apportée en introduisant une nouvelle variable, y_i^k. Cette variable a été créée dans le but de matérialiser l'affectation d'une mission à un brancardier entre les cas où le brancardier k effectue la mission i et les cas où il effectue le trajet entre la mission i et la mission j avec la variable x_{ij}^k. De même, la variable $charge_k$ a été introduite pour différencier le temps de travail et le nombre de missions (m_k) effectuées par le brancardier k. La création de ces deux variables a permis de clarifier les concepts et de lever les incompréhensions éventuelles.

Enfin, nous avons également créé la variable $p_{ST}{}_i^k$ dans le but de vérifier l'absence de sous-tournées. Cette variable nous a permis d'effectuer des contrôles supplémentaires pour garantir l'intégrité de notre modèle.

Ces ajustements et ajouts de variables ont contribué à éliminer les ambiguïtés et les malentendus potentiels, renforçant ainsi la clarté et la robustesse de notre approche.

b) Fonctions objectif

La nouvelle fonction objectif était formulée comme multicritères, c'est-à-dire :

(1) La minimisation de $D_{moyen_{ij}} \times x_{ij}^k$, qui représente la distance de trajet à vide entre deux missions i et j pour le brancardier k,
(2) La minimisation de la variable de retard, $late_i^k$,
(3) La minimisation de l'écart de charge de travail entre les brancardiers, représentée par $t_k^- + t_k^+$ pour chaque brancardier k.

Cette fonction objectif devait donc permettre de prendre en compte plusieurs critères importants, tels que la réduction des trajets à vide, la gestion des retards et l'équilibrage de la charge de travail entre les brancardiers.

$$Z = \min\left(\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=1}^{B} D_{moyen_{ij}} * x_{ij}^k + \sum_{i=0}^{N}\sum_{k=1}^{B}\left(late_i^k\right) + \sum_{k=1}^{B}(t_k^- + t_k^+)\right)$$

Équation 1: Ancienne fonction objectif

Nous avons revisité et amélioré cette fonction pour obtenir des résultats plus performants. Plus précisément, nous avons supprimé la partie de la fonction qui concerne la minimisation de l'écart de charge entre les brancardiers ainsi que la minimisation des trajets à vide. Ces

ajustements ont été réalisés dans le but d'optimiser davantage la répartition de la charge de travail entre les brancardiers. En prenant en compte cet aspect clé, nous visons à obtenir une répartition plus équilibrée des tâches, ce qui contribue à une meilleure efficacité globale du système. Grâce à ces améliorations, nous nous attendons à des résultats plus cohérents et à une meilleure utilisation des ressources disponibles. Ces deux facteurs ont été déplacés dans le champs des contraintes où ils sont encadrés par des bornes supérieurs et inférieurs.

$$Z = \min\left(\sum_{i=0}^{N}\sum_{k=1}^{B}\left(late_i^k\right)\right)$$

Équation 2: Fonction objectif

c) Contraintes

Dans le cadre de notre travail sur le modèle, nous avons effectué des modifications substantielles sur la quasi-totalité des contraintes. Ces ajustements ont été nécessaires pour aligner le modèle sur les spécifications et les exigences du projet. En révisant les contraintes existantes, nous avons pu affiner et clarifier les relations entre les variables et l'objectif du modèle. Les modifications apportées nous ont également permis de mieux tenir compte des contraintes opérationnelles et des contraintes spécifiques liées à la gestion des tournées des brancardiers.

Plusieurs contraintes ont été mises en place pour garantir la cohérence et l'efficacité de notre modèle. La contrainte (c1) veille à ce que l'heure de départ de chaque mission soit supérieure ou égale à zéro, évitant ainsi les départs négatifs. De même, la contrainte (c2) assure que le retard du brancardier k à la mission i est également supérieur ou égal à zéro, excluant les retards négatifs. Pour maintenir des retards raisonnables, la contrainte (c3) limite le retard du brancardier k à la mission i en fonction d'une valeur maximale acceptable. Les contraintes (c4) et (c5) s'assurent de la séquence logique des missions, où la mission j suit la mission i dans l'ordre. Les contraintes (c6), (c7), (c8) et (c9) gèrent le nombre de brancardiers nécessaires pour chaque mission, garantissant que les missions requérant un ou deux brancardiers soient affectées en conséquence. Les contraintes (c10) et (c11) définissent les points de départ et d'arrivée pour chaque brancardier, qui peuvent être le dépôt (mission 0) ou la salle de repos des brancardiers. La contrainte (c12) empêche une mission de boucler sur elle-même, évitant les boucles répétitives. En utilisant la formule de Miller-Tucker-Zemlin, la contrainte (c13) assure qu'il n'y a pas de sous-tournées pour chaque brancardier. La contrainte (c14) compte le nombre de missions effectuées par chaque brancardier, tandis que la contrainte (c15) limite ce nombre à un maximum de 20 pour éviter une surcharge de travail. Les contraintes (c16) et (c17) calculent et vérifient la charge de travail de chaque brancardier, en veillant à ce qu'elle respecte la capacité de travail préétablie. La contrainte (c18) garantit qu'il n'y a pas de chevauchement entre deux missions consécutives, assurant ainsi un enchaînement sans interruption. Enfin, la contrainte (c19) encadre les départs des missions en prenant en compte la tolérance du retard maximal accepté pour chaque mission, assurant une gestion adéquate des horaires de départ en fonction des contraintes temporelles. L'ensemble de ces contraintes contribue à la fiabilité et à la pertinence de notre modèle. La contrainte (c20) établit que les dates de départ (Hdep) des brancardiers k1 et k2 seront les mêmes si leurs variables d'affectation (y) sont toutes deux égales à 1. Cette condition conduit à l'égalité des dates de départ des brancardiers. La contrainte (c21) est liée à la contrainte (c13), elle stipule que si le brancardier k est affecté à une mission i (y = 1), alors la différence entre l'heure de départ (Hdep) de la mission i pour le brancardier k et le temps de trajet vers cette mission (Hrdv) doit être égale au retard (late) de la mission i pour le brancardier k. Si le brancardier n'est pas affecté à la mission i (y = 0), alors l'heure de départ (Hdep) de la mission i pour le brancardier k doit être égale au retard (late) de la mission i pour le brancardier k.

Les améliorations apportées à notre modèle ont eu un impact significatif, conduisant à une représentation plus précise et adaptée au contexte spécifique de notre projet. Ces ajustements ont considérablement amélioré la performance et la pertinence de notre modèle, nous permettant d'obtenir des résultats plus fiables et efficaces. Pour évaluer ces améliorations, nous avons procédé à des tests en utilisant le logiciel CPLEX d'IBM, reconnu pour ses capacités d'optimisation mathématique avancées. Les résultats obtenus ont confirmé l'efficacité de notre modèle amélioré, en démontrant une meilleure résolution des problèmes d'optimisation et une meilleure adéquation aux contraintes spécifiques de notre projet en comparaison aux résultats qui ont précédé. Notre modèle amélioré et l'utilisation de CPLEX ont donc été couronné de succès pour obtenir des solutions optimales et fiables pour notre problématique de gestion des tournées des brancardiers.

2. CPLEX[3]

Le logiciel CPLEX d'IBM est un outil puissant et incontournable dans le domaine de l'optimisation mathématique. Il occupe une place centrale dans la résolution de problèmes complexes dans divers secteurs tels que la logistique, la finance, la production et la planification des ressources. Grâce à ses fonctionnalités avancées et à sa grande polyvalence, CPLEX offre une bibliothèque puissante d'algorithmes de programmation linéaire, de programmation en nombres entiers et de programmation quadratique.

L'une des caractéristiques remarquables de CPLEX est sa capacité à résoudre des problèmes de programmation linéaire à grande échelle. Il peut gérer efficacement des milliers de variables et de contraintes grâce à des algorithmes sophistiqués. Cela permet aux utilisateurs de modéliser des problèmes complexes et d'obtenir des solutions précises, voire proches de l'optimum.

En outre, CPLEX peut également résoudre des problèmes de programmation en nombres entiers (MIP), ce qui est essentiel pour les problèmes de décision impliquant des variables entières ou binaires ce qui est souvent le cas en optimisation combinatoire. Cette fonctionnalité élargit considérablement la portée des problèmes pouvant être résolus avec CPLEX.

Un autre aspect important du logiciel est sa prise en charge de la programmation quadratique. Cela permet de modéliser des problèmes plus réalistes, où les objectifs et les contraintes sont formulés sous forme de fonctions quadratiques. De plus, CPLEX est capable de gérer les contraintes non linéaires grâce aux techniques d'optimisation non linéaire intégrées à sa structure.

En termes de performances, CPLEX est optimisé pour fournir des résultats rapides et précis. Il tire parti de techniques d'optimisation avancées telles que le branch and bound, les plans de coupe et les heuristiques pour accélérer la recherche de la meilleure solution possible.

Dans le cadre de notre projet, nous avons utilisé le logiciel CPLEX pour effectuer une vérification rigoureuse des modifications apportées à notre modèle. L'utilisation de CPLEX nous a permis de valider et de confirmer la validité de notre modèle mathématique final. En résolvant de manière efficace et précise les problèmes d'optimisation liés à notre projet, CPLEX a joué un rôle essentiel. Son utilisation nous a apporté une confiance accrue dans les résultats obtenus et nous a permis de prendre des décisions éclairées, basées sur des données vérifiées et des analyses approfondies.

[3] https://www.ibm.com/fr-fr/analytics/cplex-optimizer

VI. Méthode approchée : Métaheuristique

1. Présentation OptaPlanner

OptaPlanner est un moteur de planification de contraintes open-source conçu en Java selon une approche orientée objet. Il est conçu pour résoudre des problèmes d'optimisation réels comme l'affectation des ressources, la planification des horaires, la gestion de projet ou encore le problème de tournées de véhicules. OptaPlanner utilise des algorithmes d'optimisation pour trouver automatiquement des solutions quasi optimales aux problèmes, en se basant sur un système de calcul de score performant.

OptaPlanner dispose également d'une interface graphique pour choisir le problème étudié et exécuter des exemples.

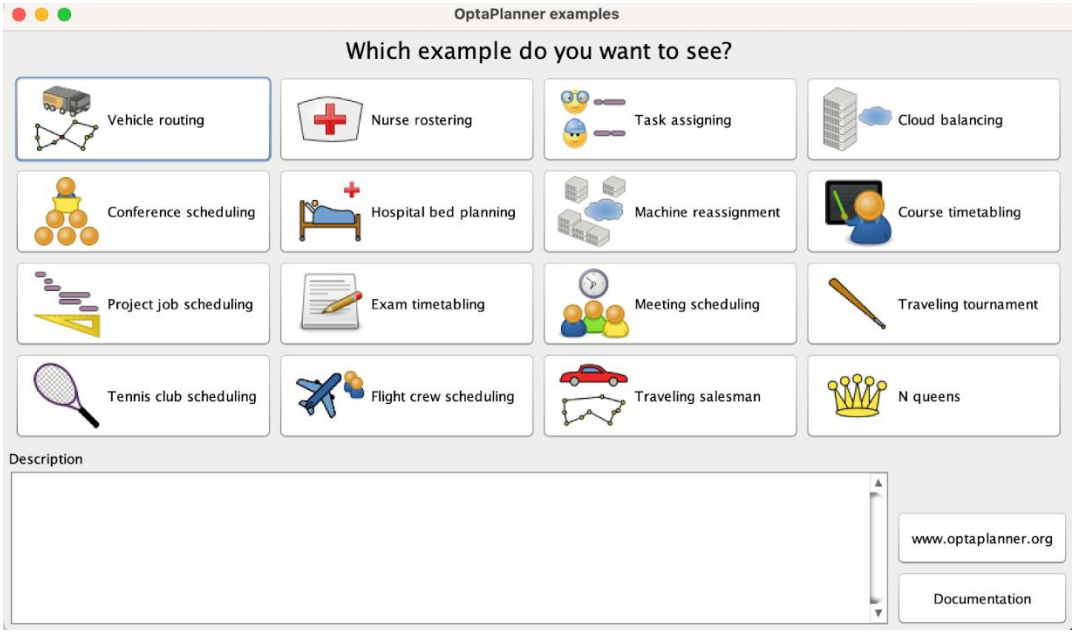

Figure 3:Présentation de l'interface d'accueil d'OptaPlanner

OptaPlanner traite donc de notre problème de VRP, en incluant la notion de capacité et de temps. Le logiciel propose des solutions à des problèmes de CVRP (Capacitated Vehicle Routing Problem) et CVRPTW (Capacitated Vehicle Routing Problem with Time Windows).

Pour la résolution des problématiques de VRP, plusieurs classes sont utilisées dans le code, nous avons identifié les plus importantes :

- **Location** : Classe abstraite représentant une position géographique. Elle sert de classe de base pour des localisations plus spécifiques comme **AirLocation** (localisation par vol d'oiseau) ou **RoadSegmentLocation** (localisation par voie terrestre). Cette classe possède des fonctions permettant de calculer une distance entre deux localisations. Ses attributs sont :
    - *name* : Nom de la localisation
    - *latitude* : Latitude de la localisation
    - *longitude* : Longitude de la localisation
    - *id* : Identifiant unique de la localisation
- **Customer** : Classe représentant un client dans le VRP, elle possède ces attributs :

- *demand* : Quantité de ressources représentée par le client
- *vehicle* : Véhicule responsable de la prestation de service au client
- *previousStandstill* : Référence à la localisation du précédent client
- *nextCustomer* : Référence à la localisation du client suivant
- *distanceFromPreviousStandstill* : Distance entre le client actuel et le précédent
- *distanceToDepot* : Distance entre le client et le dépôt

- **Vehicle** : Classe représentant un véhicule dans le VRP, possédant comme attributs :
  - *capacity : Capacité maximale de marchandises que peut transporter le véhicule*
  - *depot : Localisation du dépôt à partir duquel le véhicule effectue ses tournées*
  - *nexCustomer : Référence au client suivant dans le cheminement du véhicule*
  - *distanceToNextCustomer : Distance entre le véhicule et le client suivant*
  - *route :* Liste ordonnée des clients que le véhicule doit visiter dans sa tournée

- **Depot** : Classe représentant un dépôt dans le VRP, elle hérite de la classe **Location** donc possède des propriétés géographiques. Elle possède les attributs suivants :
  - *distanceToNextCustomer : Distance entre le dépôt et le premier client dans la tournée des véhicules*
  - *vehicleList : Liste des véhicules associés au dépôt*
  - *visitList : Liste ordonnée des clients que les véhicules associés au dépôt doivent visite*r

- **TimeWindowedCustomer** : Classe dérivée de **Customer** ajoutant des contraintes de fenêtre de temps, avec des attributs supplémentaires tels que :
  - *readyTime : Moment le plus tôt auquel le client peut être visité*
  - *dueTime : Moment le plus tard auquel le client doit être visité*
  - *arrivalTime* : Moment prévu où le véhicule arrive chez le client

- **VehicleRoutingSolution** : Classe encapsulant toutes les informations nécessaires pour résoudre un VRP :
  - *List<Customer> : Liste de tous les clients à desservir*
  - *List<Depot> : Liste des dépôts à partir desquels les véhicules peuvent commencer et terminer leurs tournées*
  - *List<Vehicle> : Liste de tous les véhicules disponibles pour effectuer les livraisons*
  - *Score : Repré*sente le score de la solution, qui évalue sa qualité

- **VehicleRoutingConstraintProvider** : Classe fournissant des méthodes de satisfaction de contraintes dans le VRP, telles que la capacité du véhicule, les distances, les horaires de livraison, etc…

2. Notion spécifique à l'outil

En plus de ces classes, OptaPlanner utilise des notions spécifiques à l'outil et aux problématiques traitées.

Parmi ces notions nous avons :

- Classe d'entité de planification (**Planning Entity**).

Lorsque nous essayons de trouver une solution à un problème, certaines parties peuvent changer au fur et à mesure du processus de recherche de la solution. Ces parties qui changent sont appelées les "variables de décision principales" dans la planification.

Dans notre modèle, nous utilisons des classes spéciales appelées **"Planning Entities"** pour représenter ces parties variables. Nous utilisons l'annotation **"@PlanningEntity"** pour marquer ces classes.

Dans certains cas d'utilisation, comme les VRP et CVRP, il est important de relier les instances de Planning Entity les unes aux autres. Cela crée une chaîne où chaque instance pointe vers une autre, ce qui permet de créer un ordre entre les entités à planifier. Chaque Planning Entity fait partie d'une chaîne qui démarre à un point de départ et se termine à un point d'arrivée. On appelle cela une "chaîne ouverte". Nous pouvons prendre pour exemple une tournée d'un véhicule, qui a comme point de départ et d'arrivée son dépôt.

- Variable de planification (**planning variable**)

Une variable de planification est une valeur dans un problème d'optimisation qui peut être modifiée par l'algorithme d'optimisation pour améliorer la solution. Elle représente une décision ou un choix que l'algorithme peut ajuster pour trouver la meilleure configuration possible. Par exemple, dans un VRP, une variable de planification pourrait être l'affectation d'un client à un véhicule, ou l'ordre de visite des clients par un véhicule.

L'algorithme d'optimisation examine différentes combinaisons de valeurs pour ces variables de planification afin de rechercher la configuration optimale qui respecte les contraintes et maximise ou minimise l'objectif donné. Il modifie les variables de planification en appliquant des opérations spécifiques, telles que l'échange de clients entre les véhicules ou le déplacement d'une visite à un autre emplacement dans la tournée.

Les variables de planification sont donc des éléments que l'algorithme d'optimisation ajuste pour trouver la meilleure solution dans un problème donné. Elles représentent les décisions ou les choix qui peuvent être modifiés afin d'optimiser l'objectif recherché.

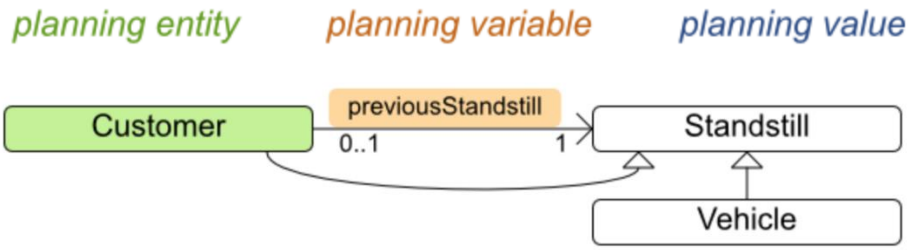

Figure 4: schéma des variables de plannification

Dans cette figure, on a des entités clés comme les clients (customers), les positions précédentes (previousStandstill) et les véhicules.

Chaque client représente une étape ou une visite potentielle dans une tournée de véhicules. Les clients ont des attributs tels que leur emplacement, leur demande (par exemple, le nombre de colis à livrer) et leur fenêtre de temps disponible.

La variable de planification (planning variable) "previousStandstill" est associée à chaque client. Elle représente la position précédente du client dans la tournée. Par exemple, si le client A est visité avant le client B, la variable "previousStandstill" du client B pointera vers le client A.

Les véhicules sont également des entités de planification. Ils sont associés à la variable de planification "standstill" (position actuelle) qui représente la position actuelle du véhicule dans la tournée. Le "standstill" peut être soit un client, soit le dépôt initial (départ et arrivée).

Ainsi, les liens entre ces entités sont les suivants : un client est lié à sa position précédente (previousStandstill) dans la tournée, et le véhicule est lié à sa position actuelle (standstill) dans la tournée. Ces relations permettent de définir l'ordre des clients visités par chaque véhicule et de construire les tournées de manière cohérente.

Par exemple, pour une tournée avec 4 clients et 2 véhicules, les valeurs des variables changent selon l'état d'avancement de la tournée :

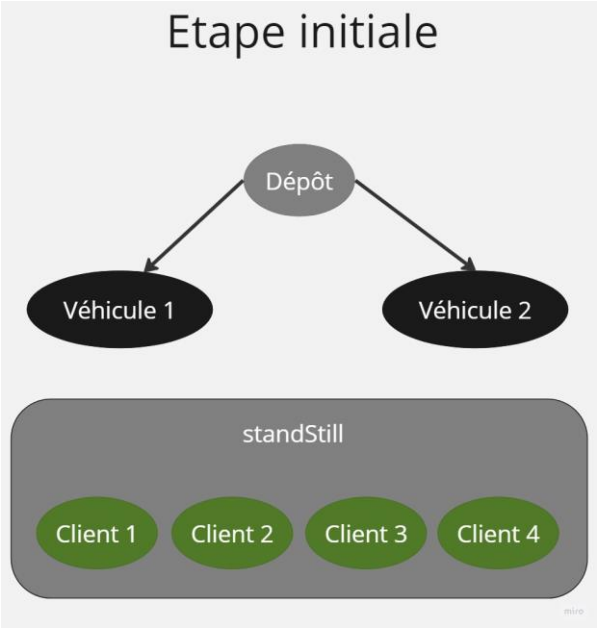

Figure 5: Étape initiale exemple

- Étape initiale : les variables previousCustomer et nextCustomer sont null

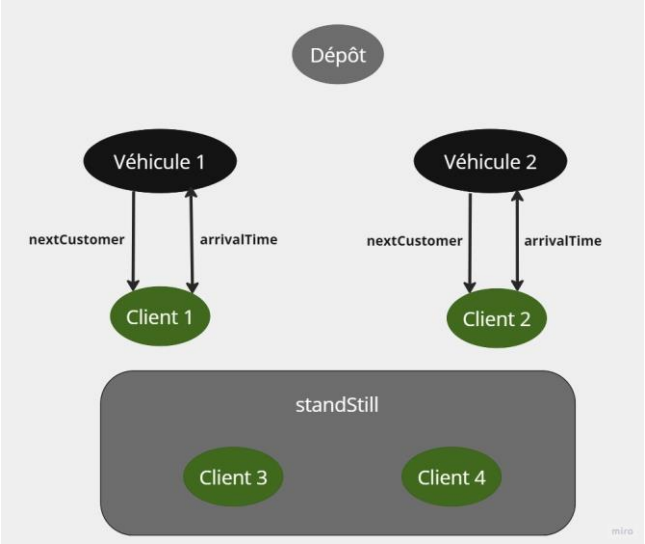

Figure 6: Étape 1 exemple

- Étape 1:
  - previousCustomer: null
  - nextCustomer: pour V1 = C1, pour V2 = C2

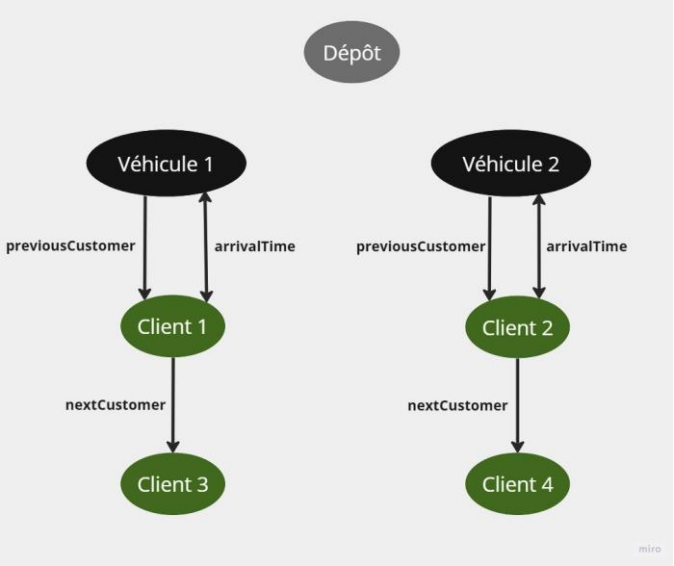

Figure 7: Étape 2 exemple

- Étape 2:
  - previousCustomer: pour V1 = C1, pour V2 = C2
  - nextCustomer: pour V1 = C3, pour V2 = C4

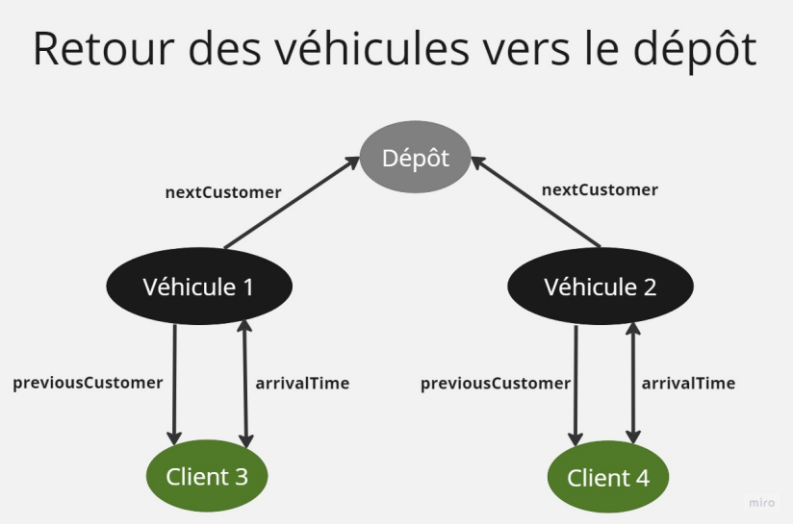

Figure 8: Étape 3 exemple

- Étape 3:
  - previousCustomer: pour V1 = C3, pour V2 = C4
  - nextCustomer : pour V1 = Dépôt, pour V2 = Dépôt
- Variable d'ombre (**Shadow variable**)

Dans OptaPlanner, il existe un concept de "variable d'ombre" qui est une variable secondaire liée à une variable principale. La valeur de la variable d'ombre dépend de manière mathématique de la variable principale. En d'autres termes, il y a une relation de cause à effet pour calculer la valeur de la variable d'ombre à partir de la variable principale.

Une variable d'ombre est une variable de planification dont la valeur correcte peut être déduite à partir de l'état des variables principales. Par exemple, dans un VRPTW, l'heure d'arrivée d'un véhicule chez un client (variable d'ombre) peut être calculée en fonction des clients précédemment visités par ce véhicule (variable principale). Lorsque les clients pour un véhicule changent, l'heure d'arrivée prévue pour chaque client est automatiquement ajustée.

OptaPlanner optimise principalement les variables principales, mais s'assure que les variables d'ombre correspondantes sont mises à jour correctement lorsque les variables principales changent.

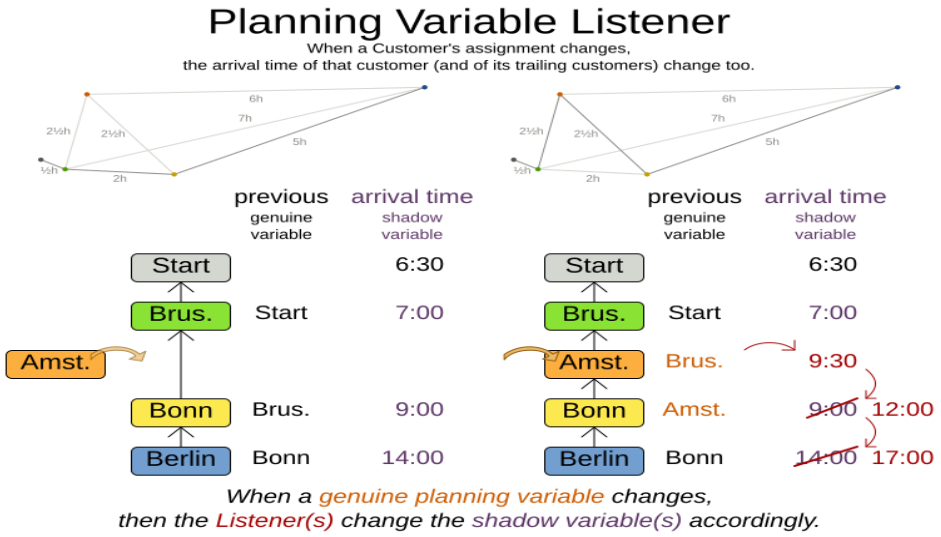

Figure 9: Figure présentant différents trajets

Sur cette figure on peut voir différents trajets autour des capitales avec les horaires qui sont précisés. On a sur la partie gauche les horaires prévus des différents arrêts tandis que sur la droite ce sont les horaires qui ont été réalisés. Ici les variables d'ombres (shadows variables) sont les retards, les horaires qui ont dû être modifiés par rapport à ce qui était prévu.

- La notion de chaînage (chaining) et ancrage (anchoring)

Dans OptaPlanner, la notion de "chainage" se réfère à la manière dont les entités sont organisées dans une séquence ou une chaîne, généralement dans un ordre spécifique. Prenons l'exemple d'un problème de tournées de véhicules. Supposons que nous ayons plusieurs véhicules et chaque véhicule doit visiter plusieurs clients. Dans ce cas, le chaînage est utilisé pour établir l'ordre de visite des clients par chaque véhicule. Chaque client est une entité et est connecté au client précédent dans la séquence de visite du véhicule. Cela crée une chaîne de clients reliés les uns aux autres, définissant ainsi l'ordre des visites pour chaque véhicule.

La notion d'ancrage se réfère à la fixation d'une entité spécifique dans un contexte particulier. Reprenons l'exemple précédent des tournées de véhicules. Supposons que chaque véhicule ait un point de départ fixe, appelé le dépôt. Dans ce cas, le dépôt est ancré à la première position de chaque tournée de véhicule. Cela signifie que le dépôt est lié de manière stable à la première entité de chaque tournée, et il ne peut pas être déplacé ou réarrangé lors de l'optimisation. L'ancrage du dépôt garantit que chaque tournée de véhicule commence par le dépôt, ce qui est une exigence fondamentale du problème.

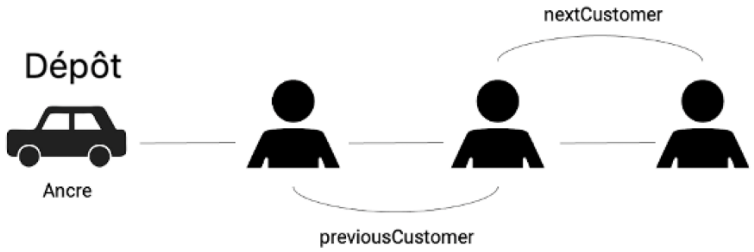

Figure 10: Schéma explicatif des ancres

3. Spécification des contraintes

Dans OptaPlanner, une contrainte est une règle ou une condition qui doit être respectée lors de la résolution d'un problème d'optimisation. Les contraintes définissent les conditions auxquelles les solutions doivent se conformer pour être valides.

Ces contraintes peuvent être catégorisées en trois types : hard, soft et medium. Chaque type de contrainte joue un rôle différent dans le processus d'optimisation.

- Contraintes **Hard** (Dures) : Les contraintes hard sont des contraintes qui doivent absolument être respectées. Elles définissent les règles fondamentales et incontournables du problème. Si une solution viole une contrainte hard, elle est considérée comme invalide et sera pénalisée de manière significative. Par exemple, dans le cas des tournées de véhicules, une contrainte hard pourrait être la capacité maximale de chargement des véhicules. Si un véhicule dépasse sa capacité, la solution sera considérée comme invalide et pénalisée de manière importante.
- Contraintes **Medium** (Intermédiaires) : Les contraintes medium sont un compromis entre les contraintes hard et soft. Elles ont une importance intermédiaire et peuvent être traitées comme des contraintes soft ou hard en fonction du contexte du problème. Les violations de contraintes medium sont pénalisées, mais de manière moins sévère que les contraintes hard. Les contraintes medium offrent une flexibilité dans le processus d'optimisation. Elles permettent de spécifier des critères intermédiaires qui peuvent influencer les décisions prises par le solveur.
- Contraintes **Soft** (Souples) : Les contraintes soft sont des contraintes qui sont souhaitables à respecter, mais qui peuvent être violées si nécessaire pour obtenir une meilleure solution globale. Contrairement aux contraintes hard, les violations de contraintes soft ne rendent pas la solution invalide, mais elles sont tout de même pénalisées. Les contraintes soft permettent de trouver un compromis entre les différentes mesures de performance. Par exemple, dans le cas des tournées de véhicules, une contrainte soft pourrait être l'optimisation de la distance totale parcourue. Bien que cette contrainte soit souhaitable, il peut être acceptable de la violer légèrement si cela permet d'optimiser d'autres aspects, comme la réduction du temps d'attente des clients.

Pour construire une contrainte, il existe plusieurs types de faisabilité. En effet l'une d'entre elle se trouve dans la classe "**VehicleRoutingConstraintProvider**". Celle-ci est codée à partir du langage Java. Nous expliquerons comment ces contraintes sont utilisées dans ce type de langage dans les paragraphes suivants. Cependant il existe une autre façon de faire qui est bien plus simple et qui utilise le langage DRL (Drools Rule Language) de Drools pour coder les contraintes.

Ce choix permet de déclarer les contraintes sous forme de règles qui sont séparées du modèle, ce qui permet de facilement ajouter, modifier ou même supprimer des contraintes.

Drools fournit aussi le support du calcul incrémental des scores caractérisant l'optimalité d'une solution, sans code supplémentaire. La déclaration de contrainte en Drools se compose de deux parties en logique de premier ordre : la condition « when » et la conséquence « then ».

Dans la partie « when », nous spécifions la condition déclenchant la contrainte et déclarons les paramètres en utilisant la syntaxe « $ + nom ».

Dans la partie « then », nous déclarons les éléments nécessaires en vue du calcul de score.

Voici une des contraintes que nous avons pu coder qui est celle de respecter au mieux la charge de travail maximale d'un brancardier qui effectue une tournée :

```
// ############################################################################
// Soft constraints
// ############################################################################

rule "maxWorkload"
    when
        $vehicle : Vehicle($maxWorkload : maxWorkload)
        accumulate(
            Customer(
                vehicle == $vehicle,
                $demand : demand
            );
            $totalDemand : sum($demand);
            $totalDemand > $maxWorkload
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $maxWorkload - $totalDemand);
end
```

Figure 11: Aperçu du code d'une soft contraintes

La règle "**maxWorkload**" vérifie si la somme des demandes des clients assignés à un véhicule dépasse la charge maximale de travail du véhicule. La variable *$vehicle* représente le véhicule en question, et *$maxWorkload* est sa charge maximale de travail. L'accumulateur "*accumulate*" calcule la somme des demandes de tous les clients assignés au véhicule tandis que la variable *$totalDemand* représente la somme totale des demandes des clients assignés au véhicule. Si la somme des demandes dépasse la charge maximale de travail, une pénalité est ajoutée au score en utilisant la méthode "*scoreHolder.addHardConstraintMatch(kcontext, $maxWorkload - $totalDemand);*".

Cette règle permet donc de favoriser les solutions où la charge de travail maximale des brancardiers est respectée au mieux, en ajoutant des pénalités au score chaque fois que cette contrainte est violée.

4. La fonction objectif

La fonction objectif est présente dans le code d'OptaPlanner, mais sous une version différente. Son but est d'évaluer chaque tournée et d'y attribuer un score. En simulant une multitude de tournées, OptaPlanner calcule leurs scores en tenant compte des différentes contraintes et met à jour l'affichage jusqu'à la tournée que l'outil juge optimale, donc celle ayant le score le plus élevé.

Cette fonction se trouve dans la classe **VehicleRoutingEasyScoreCalculator**, sous le nom de "calculateScore" :

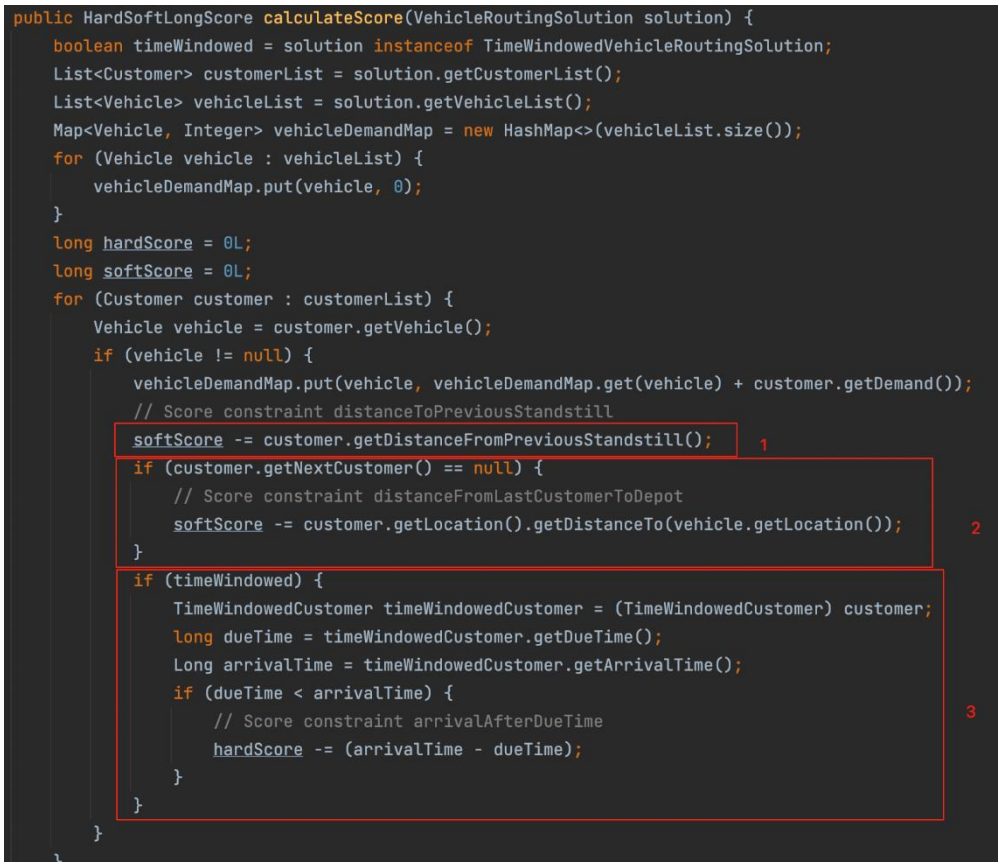

Figure 12: Aperçu 1 de la fonction objectif

Figure 13: Aperçu 2 de la fonction objectif

La fonction commence par initialiser certaines variables nécessaires au calcul du score. Elle parcourt ensuite la liste des Customer présents dans la solution. Pour chaque Customer, elle récupère le véhicule qui lui est assigné et met à jour le cumul de la demande du véhicule correspondant. Elle calcule ensuite le score partiel en fonction de différentes contraintes du problème :

1. Contrainte **distanceToPreviousStandstill** : Elle soustrait la distance entre le client actuel et son précédent client dans la tournée, contribuant au soft score
2. Contrainte **distanceFromLastCustomerToDepot** : Si le client actuel est le dernier client de la tournée, elle soustrait la distance entre ce client et le dépôt, également contribuant au soft score
3. Contrainte **arrivalAfterDueTime** : Si le problème est basé sur des fenêtres de temps (CVRPTW), elle vérifie si le client est arrivé après l'heure limite fixée (dueTime) et, le cas échéant, soustrait la différence entre l'heure d'arrivée et l'heure limite au hard score

4. Contrainte **vehicleCapacity** : La fonction parcourt le map vehicleDemandMap contenant chaque véhicule et sa demande cumulative. Elle vérifie si la demande dépasse la capacité du véhicule et soustrait l'excédent au hard score

Enfin, la fonction retourne le score total calculé, composé du hard et soft score, encapsulés dans un objet de type HardSoftLongScore.

5. Solution envisagée

Pour utiliser OptaPlanner dans nos tests de tournées de brancardiers, nous avons adopté une approche consistant à simuler les jeux de données utilisés sur CPLEX dans OptaPlanner.

Ces jeux de données représentent nos missions de brancardage, comprenant les informations essentielles telles que les horaires de rendez-vous, la durée des missions et le nombre de brancardiers demandés.

Ensuite, nous avons développé des contraintes personnalisées qui reflètent les exigences et les limitations propres à notre contexte. Par exemple, nous avons pris en compte la capacité maximale des brancardiers, les contraintes de disponibilité temporelle et les temps de trajet estimés entre les missions.

En combinant ces données simulées, nos contraintes spécifiques et les contraintes propres à OptaPlanner, nous avons configuré l'outil pour résoudre le problème de tournées de brancardiers de manière adaptée.

6. Problèmes rencontrés

Lors de nos tests de tournées de brancardiers avec OptaPlanner, nous avons rencontré certains problèmes liés à la nature du problème que nous souhaitions résoudre. En effet, OptaPlanner est principalement conçu pour résoudre des problèmes de CVRP, tandis que notre problème de tournées de brancardiers ne tient pas compte de la capacité,

L'un des principaux problèmes que nous avons rencontrés est que la notion de charge et de demande des clients dans OptaPlanner n'était pas directement adaptable à notre problème. Dans notre contexte, chaque mission de brancardage est associée à un patient, qu'il transporte d'un point A à un point B. Cependant, lorsque nous avons essayé de modéliser ce transport comme une demande ou une charge dans OptaPlanner, une erreur est survenue car cela ne correspondait pas à la structure prévue pour les problèmes de CVRP gérés par OptaPlanner.

Un autre problème auquel nous avons été confrontés était l'absence de coordonnées dans nos jeux de données de tournées de brancardiers. Les coordonnées spatiales sont généralement utilisées dans les problèmes de VRP pour calculer les distances entre les lieux de prise en charge et de dépôt. Dans notre cas, nous avons dû nous appuyer sur des jeux de données d'exemple fournis avec OptaPlanner, qui contenaient des coordonnées fictives, afin de pouvoir estimer les distances entre les missions de brancardage. Nous avons ensuite adapté ces jeux de données en intégrant nos contraintes de temps propres aux tournées de brancardiers. Cela a introduit une certaine approximation dans nos résultats, car les coordonnées ne correspondaient pas exactement à notre environnement réel.

Malgré ces problèmes, nous avons pu contourner ces limitations en adaptant notre modélisation et en faisant des compromis pour résoudre partiellement notre problème de tournées de brancardiers avec OptaPlanner. Cependant, il est important de noter que ces limitations inhérentes à l'outil nous ont rappelé que chaque problème de tournées a ses particularités et qu'une adaptation et une personnalisation soigneuses sont nécessaires pour obtenir des résultats optimaux dans des contextes spécifiques comme le nôtre.

7. Développement mis en place

Nous avons développé deux contraintes, basées sur le modèle mathématique, accompagnant les contraintes déjà implémentées sur OptaPlanner :

```
// Contrainte n°8
protected Constraint vehicleNotReturnToSameCustomer(ConstraintFactory factory) {
    return factory.forEach(Customer.class) UniConstraintStream<Customer>
            .filter(customer -> customer.getNextCustomer() != null)
            .penalize(HardSoftLongScore.ONE_HARD, Customer::getDistanceFromNextStandstillHard)
            .asConstraint( s: "vehicleNotReturnToSameCustomer");
}

// Contrainte n°7
protected Constraint vehicleReturnToDepot(ConstraintFactory factory) {
    return factory.forEach(Customer.class) UniConstraintStream<Customer>
            .filter(customer -> customer.getVehicle() != null)
            .penalize(HardSoftLongScore.ONE_HARD, Customer::vehicleHasDepot) UniConstraintBuilder<C
            .asConstraint( s: "vehicleHasDepot");
}
```

Figure 14: Aperçu du code des contraintes

Ces deux contraintes sont de type “hard”, car elles sont nécessaires au bon fonctionnement de notre solution.

La première contrainte consiste à empêcher un véhicule de retourner voir le même client après l’avoir déjà visité. Cela renforce la diversité des tournées et évite les boucles inutiles.

La deuxième contrainte permet de nous assurer que les véhicules retournent au dépôt à la fin de leurs tournées. Cette contrainte est essentielle pour que le retour au dépôt soit pris en compte lors de l’évaluation de la solution.

Pour intégrer nos données initiales utilisées sur CPLEX dans OptaPlanner, nous les avons adaptées en utilisant un fichier JSON adapté au format de données d'OptaPlanner. Nous avons reporté les fenêtres de temps pour chaque mission. Le logiciel raisonne en termes de minutes multiplié par 1000 pour le temps, afin d’éviter les erreurs d’arrondi. Par exemple, pour mettre dans la variable “readyTime” l’heure du début de mission qui est 08h50, nous devons convertir cet horaire en minutes puis multiplier par 1000 ce qui nous donne 530000.

```
"@type" : "timeWindowed",
"@id" : "TimeWindowedCustomer#2",
"id" : 2,
"location" : "AirLocation#2",
"demand" : 30,
"vehicle" : null,
"previousCustomer" : null,
"nextCustomer" : null,
"readyTime" : 530000,
"dueTime" : 539000,
"serviceDuration" : 9000,
"arrivalTime" : null,
"departureTime" : null,
"arrivalBeforeReadyTime" : false,
"arrivalAfterDueTime" : false
```

Figure 15: Variables de temps

Cependant, nous avons rencontré une limitation avec les variables “demand” et de coordonnées de latitude et de longitude de nos missions. Ces attributs spécifiques à notre problème n'étaient pas disponibles dans les données initiales utilisées sur CPLEX. Par conséquent, nous avons dû utiliser des exemples de jeux de données OptaPlanner pour ces variables manquantes.

VII. Présentation des résultats

1. Programmation linéaire

1) Comparaison avec les anciens résultats

a) Temps

Le passage de l’ancien modèle au nouveau modèle a entraîné une amélioration significative en termes de temps de calcul. Alors que l'ancien modèle prenait plusieurs heures pour générer les résultats, le nouveau modèle a réussi à produire les mêmes résultats en moins de 30 secondes. Cette réduction drastique du temps de calcul a considérablement optimisé l'efficacité de notre processus d'optimisation. Non seulement cela nous permet de gagner du temps de calcul, mais cela nous offre également une plus grande flexibilité pour effectuer des analyses et des itérations supplémentaires. Grâce à cette amélioration, nous pouvons maintenant générer plus rapidement des solutions de qualité pour des problèmes complexes, ce qui nous aide à prendre des décisions plus éclairées et à répondre aux besoins de manière plus efficace.

b) Résultats

Nous avons réalisé des expérimentations avec un jeu de données composé de 80 missions, en utilisant différents nombres de brancardiers tels que 12, 10 et 9. Les résultats obtenus dans ces configurations ont été satisfaisants, démontrant l'efficacité de notre modèle dans la gestion de ces missions. Cependant, lorsque nous avons réduit le nombre de brancardiers à 8 pour les mêmes 80 missions, nous avons constaté que le modèle atteignait ses limites. Ils constituent également un point de départ pour de futures améliorations afin de mieux gérer les scénarios complexes avec un nombre limité de ressources.

2) Comparaison de plusieurs jeux de données

Nous avons utilisé un jeu de données comprenant 80 missions pour notre étude. Parmi ces missions, 57 d'entre elles nécessitent l'intervention d'un seul brancardier, tandis que les autres missions exigeaient la présence de deux brancardiers. Nous avons effectué plusieurs exécutions du jeu de données en utilisant différents nombres de brancardiers, à savoir 12, 10 et 9. Cette approche nous a permis d'évaluer les performances et l'efficacité de notre modèle d'optimisation en fonction du nombre de ressources disponibles.

a) Indicateurs

Indicateurs retard

Nombre de brancardier	12	10	9
Retard cumulé	0	0	6

Tableau 2: indicateurs de retard en minutes

Lorsqu'il y a 12 ou 10 brancardiers disponibles, le système est conçu de manière à éviter tout retard. Grâce à une répartition optimale des missions et à une gestion efficace des ressources, les départs sont planifiés de manière à respecter les horaires et à minimiser les attentes. Cependant, avec seulement 9 brancardiers, on observe un retard de 6 minutes. Cela peut être dû à une capacité de travail légèrement inférieure par rapport à la demande, ce qui nécessite une légère adaptation dans la planification pour maintenir une performance satisfaisante.

Malgré ce léger retard, le système reste globalement fiable et garantit une exécution efficace des missions.

Indicateur sur la charge de travail temporel par brancardier

Nombre de brancardier	12	10	9
Charge moyenne	180	211	228
Charge maximale	238	308	329
Charge minimale	113	159	135
Ecart moyen	30	33	36

Tableau 3: Indicateur sur la charge de travail par brancardier temporel en minutes

En analysant la charge de travail des brancardiers dans le contexte temporel, nous constatons des variations significatives en fonction du nombre de brancardiers disponibles. Avec 12 brancardiers, la charge moyenne s'élève à 180 unités, tandis que la charge maximale atteint 238 unités et la charge minimale descend à 113 unités. L'écart moyen entre la charge maximale et minimale est de 30 unités, reflétant une répartition relativement équilibrée des tâches entre les brancardiers.

Lorsque le nombre de brancardiers est réduit à 10, la charge moyenne augmente légèrement à 211 unités, avec une charge maximale de 308 unités et une charge minimale de 159 unités. L'écart moyen entre la charge maximale et minimale s'élève à 33 unités, indiquant une légère augmentation des variations de charge entre les brancardiers.

Enfin, avec seulement 9 brancardiers, la charge moyenne atteint 228 unités, tandis que la charge maximale augmente à 329 unités et la charge minimale diminue à 135 unités. L'écart moyen entre la charge maximale et minimale est de 36 unités, témoignant d'une répartition plus inégale des tâches et d'une charge de travail plus importante pour certains brancardiers.

Indicateur sur la charge de travail par brancardier en nombre de mission

Nombre de brancardier	12	10	9
Charge moyenne	8,5	10	11
Charge maximale	11	13	14
Charge minimale	5	6	9
Ecart moyen	1,7	2,1	1,2

Tableau 4: Indicateur sur la charge de travail par brancardier en nombre de mission

En analysant la charge de travail des brancardiers en fonction du nombre de missions effectuées, nous observons des variations significatives. Avec 12 brancardiers, la charge moyenne est de 8,5 missions, tandis que la charge maximale s'élève à 11 missions et la charge minimale atteint 5 missions. L'écart moyen entre la charge maximale et minimale est de 1,7 missions, ce qui indique une répartition relativement équilibrée des missions entre les brancardiers.

Lorsque le nombre de brancardiers est réduit à 10, la charge moyenne augmente à 10 missions, avec une charge maximale de 13 missions et une charge minimale de 6 missions. L'écart moyen entre la charge maximale et minimale s'élève à 2,1 missions, ce qui témoigne d'une augmentation des variations de charge entre les brancardiers.

Enfin, avec seulement 9 brancardiers, la charge moyenne atteint 11 missions, tandis que la charge maximale augmente à 14 missions et la charge minimale atteint 9 missions. L'écart moyen entre la charge maximale et minimale est de 1,2 missions, indiquant une répartition relativement équilibrée des missions malgré le nombre réduit de brancardiers.

b) *Modélisation graphique*

Nous avons utilisé l'outil Figma pour représenter visuellement notre système de tournées. Dans notre représentation, chaque couleur est associée à un brancardier spécifique, ce qui nous permet de distinguer facilement les différentes tournées effectuées par chaque brancardier. Cette approche de visualisation nous offre une meilleure visibilité et compréhension de la répartition des missions entre les brancardiers. Nous pouvons clairement identifier les itinéraires individuels de chaque brancardier, ce qui facilite la détection d'éventuels chevauchements ou déséquilibres dans la répartition des charges de travail. Grâce à cette modélisation graphique, nous avons pu optimiser et ajuster nos plans de tournées de manière plus précise et efficace, en tenant compte des contraintes et des besoins spécifiques de chaque brancardier. La visualisation claire et distincte des tournées sur Figma a grandement contribué à l'amélioration de notre processus de planification et de gestion des ressources.

Nous avons fait le choix de ne pas synchroniser les missions à 2 brancardiers car la modélisation devenait illisible et perdait en clarté.

Modélisation de la tournée à 12 brancardiers

Nous pouvons observer grâce à la modélisation que la charge n'est pas bien répartie entre les brancardiers mais nous pouvons aussi remarquer qu'il n'y a pas de sous tournée et que les missions à deux brancardiers ont bien deux brancardiers et, de même, pour ceux nécessitant 1 brancardier.

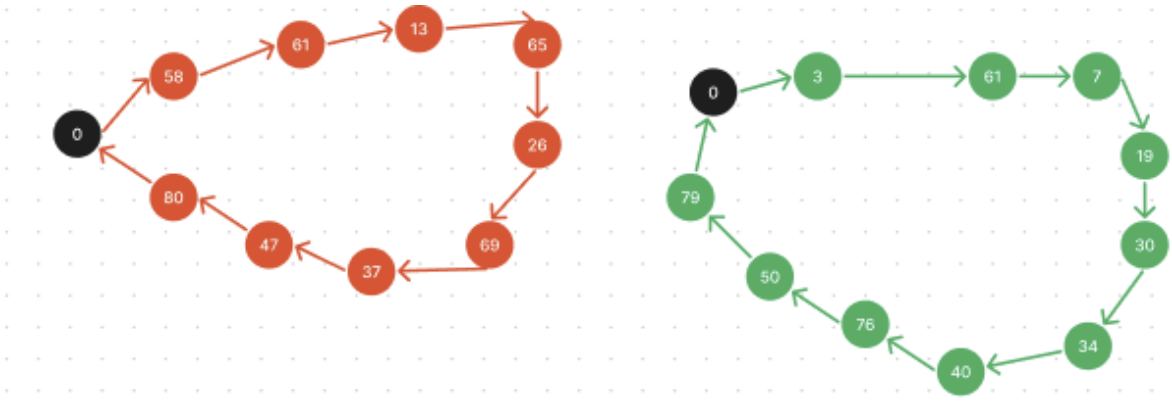

Figure 16: Aperçu des tournées des brancardiers 1 à 2

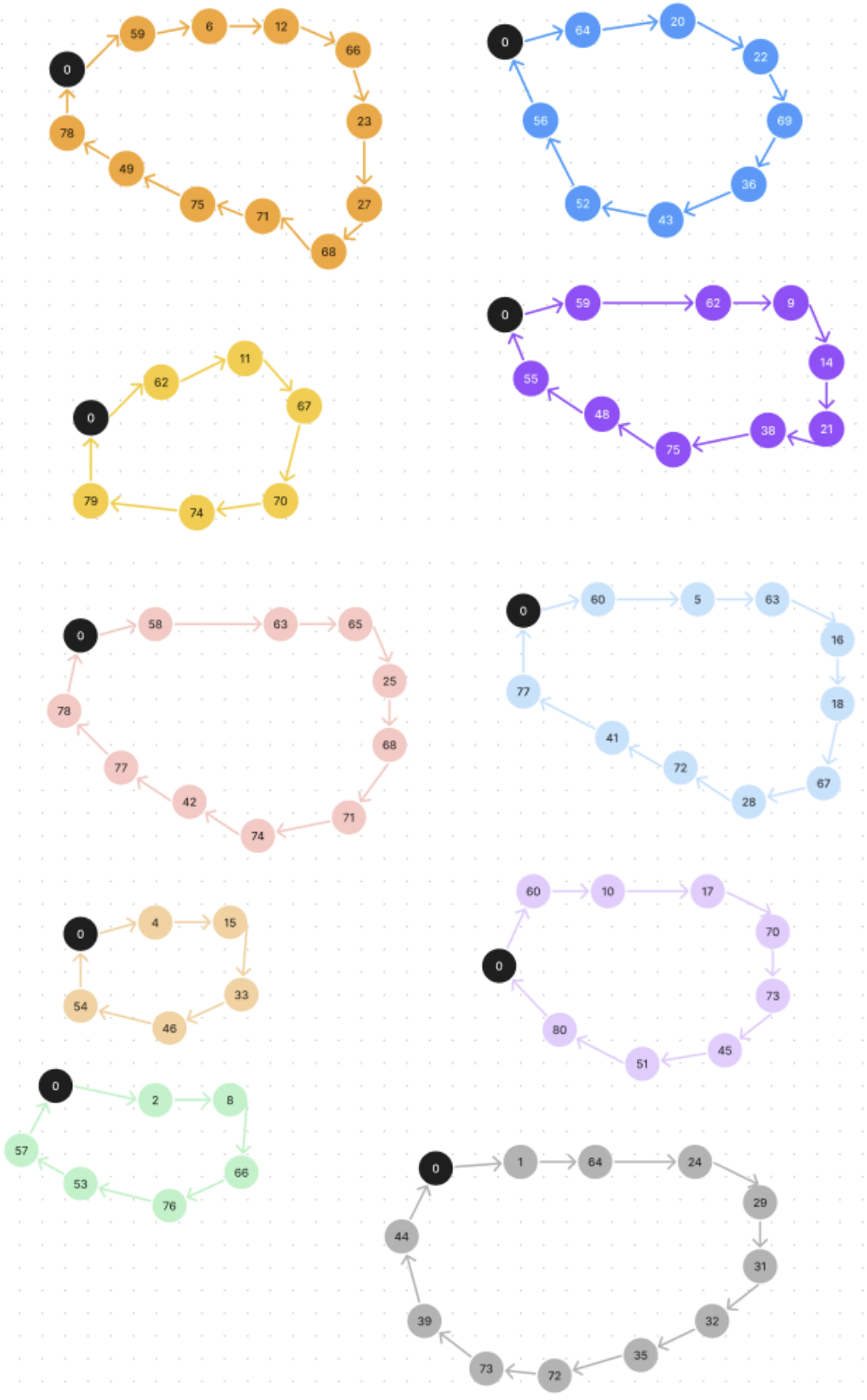

Figure 17: Aperçu des tournées des brancardiers 3 à 12

Modélisation de la tournée à 10 brancardiers

Nous pouvons faire les mêmes constatations que pour la modélisation à 12 brancardiers.

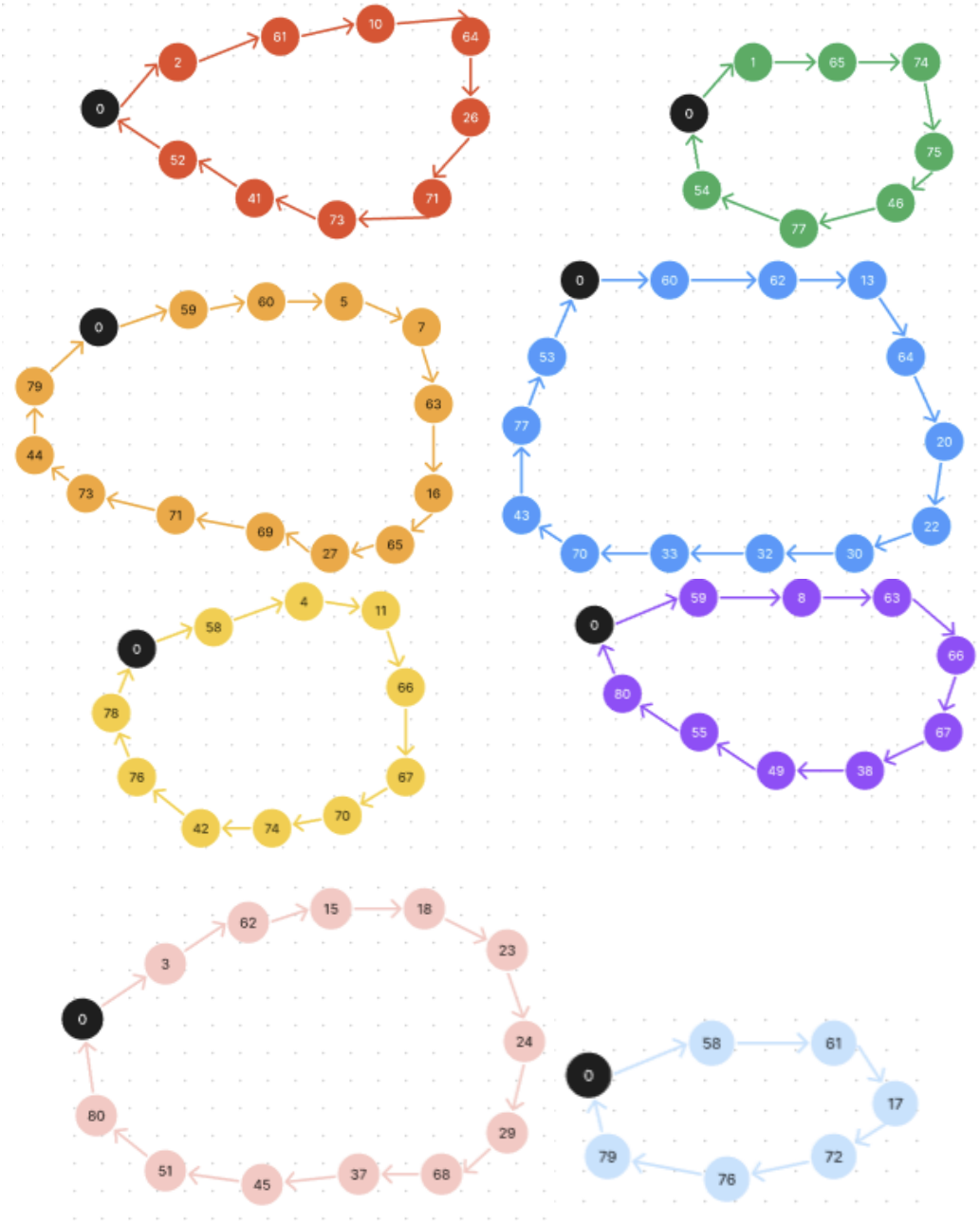

Figure 18: Aperçu des tournées des brancardiers 1 à 8

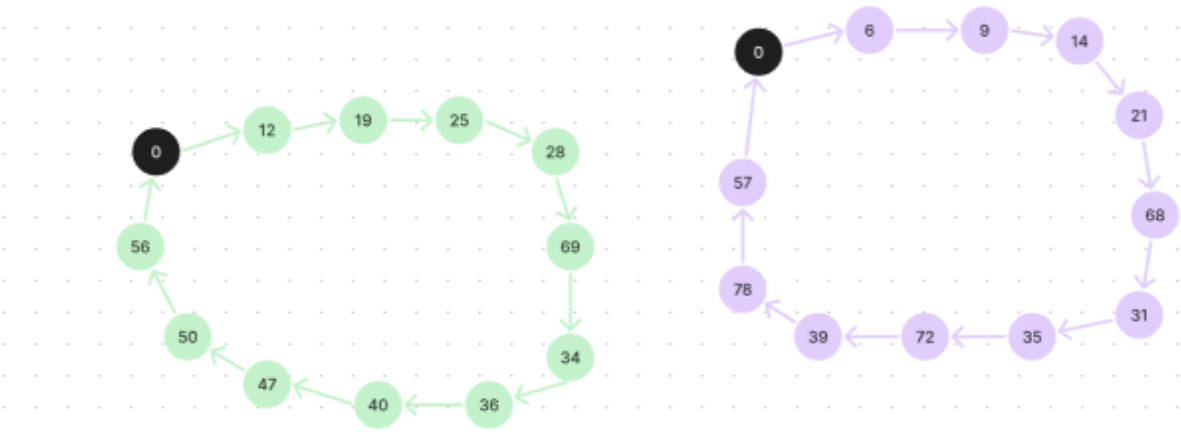

Figure 19: Aperçu des tournées des brancardiers 9 à 10

Modélisation de la tournée à 9 brancardiers

La modélisation à 9 brancardiers nous a permis de remarquer que la charge des brancardiers est plus équitables dans cette modélisation.

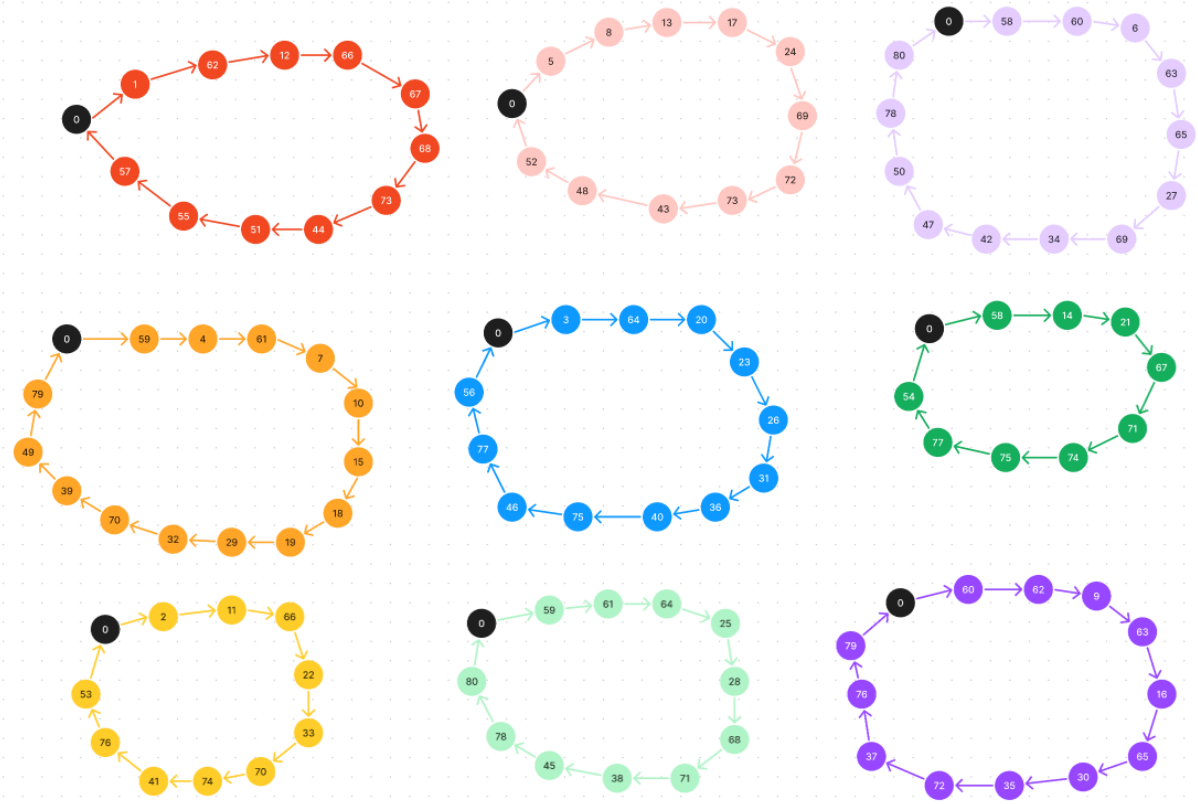

Figure 20: Aperçu des tournées des brancardiers 1 à 9

2. Optaplanner

1) Résultats sur plusieurs jeux de données

Afin d'évaluer notre modèle et nos résultats, nous avons importé les mêmes jeux de données utilisés sur CPLEX dans OptaPlanner. Ces données consistent en 80 missions, où chaque mission est associée à une fenêtre de temps pendant laquelle elle peut être exécutée, ce qui indique les contraintes de disponibilité ou de délai.

De plus, chaque mission a une durée spécifique, qui représente le temps nécessaire pour accomplir cette tâche. Cette durée est un facteur important dans la résolution du problème, car elle affecte la séquence et la répartition des missions entre les ressources disponibles.

En utilisant ces données importées, OptaPlanner cherche à trouver la meilleure répartition des missions dans le temps et entre les ressources afin d'optimiser certains objectifs spécifiques tels que la minimisation du temps total de trajet, la maximisation de la capacité d'exécution des missions, ou encore la réduction de la distance parcourue.

L'objectif final est de générer un plan de mission optimal qui respecte les contraintes de disponibilité temporelle, minimise les retards et les temps d'attente, et optimise les performances globales du système.

La différence entre les jeux de données implantés sur CPLEX et sur OptaPlanner est que sur ce dernier, nous avons la notion de capacité pour chaque véhicule, que nous n'avons pas pu modifier. De plus, les coordonnées des missions utilisées dans nos jeux de données sont basées sur d'anciens exemples de CVRPTW, donc la localisation des missions est une métrique à revoir.

Dans notre premier exemple, nous avons testé un jeu de données comportant 80 missions et 10 véhicules :

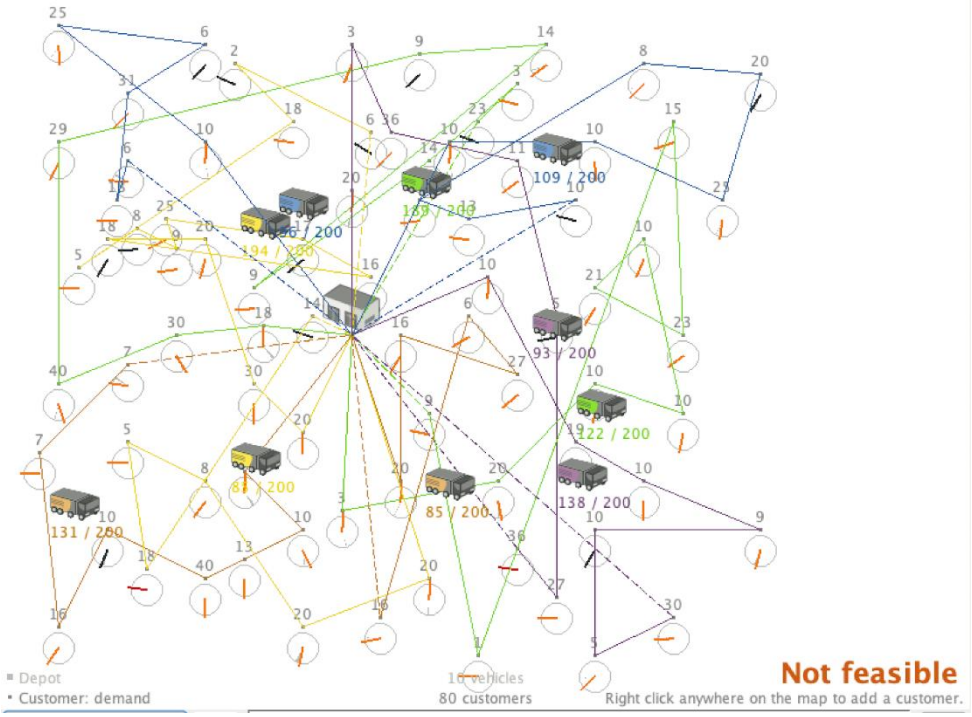

Figure 21: Résultat des tournées à 80 missions et 10 véhicules

Nous pouvons voir que ce jeu de données n'est pas faisable selon OptaPlanner, cela est due à une contrainte de type "hard" qui n'est pas respectée.

Constraint name	Constraint weight	Match count	Score
arrivalAfterDueTime	-1hard	2	-7556hard
distanceFromLastCustomerToDepot	-1soft	10	-250317soft
distanceToPreviousStandstill	-1soft	80	-1186346soft

Constraint matches of selected constraint type

[TimeWindowedCustomer-44] = -7153hard
[TimeWindowedCustomer-48] = -403hard

Figure 22: Aperçu des contraintes non respectées

La contrainte “arrivalAfterDueTime” n’a pas été respectée 2 fois comme nous pouvons le voir dans la colonne « Match Count », sur les missions 44 et 48, elle consiste à pénaliser les tournées où des véhicules arrivent en retard à des missions. Étant une contrainte “hard”, le CVRPTW est automatiquement considéré comme non faisable.

En partant du principe qu’on tolère les retards, si nous modifions la sévérité de cette contrainte en la rendant “soft” (donc prise en compte dans le calcul de la performance des tournées), nous avons ce résultat :

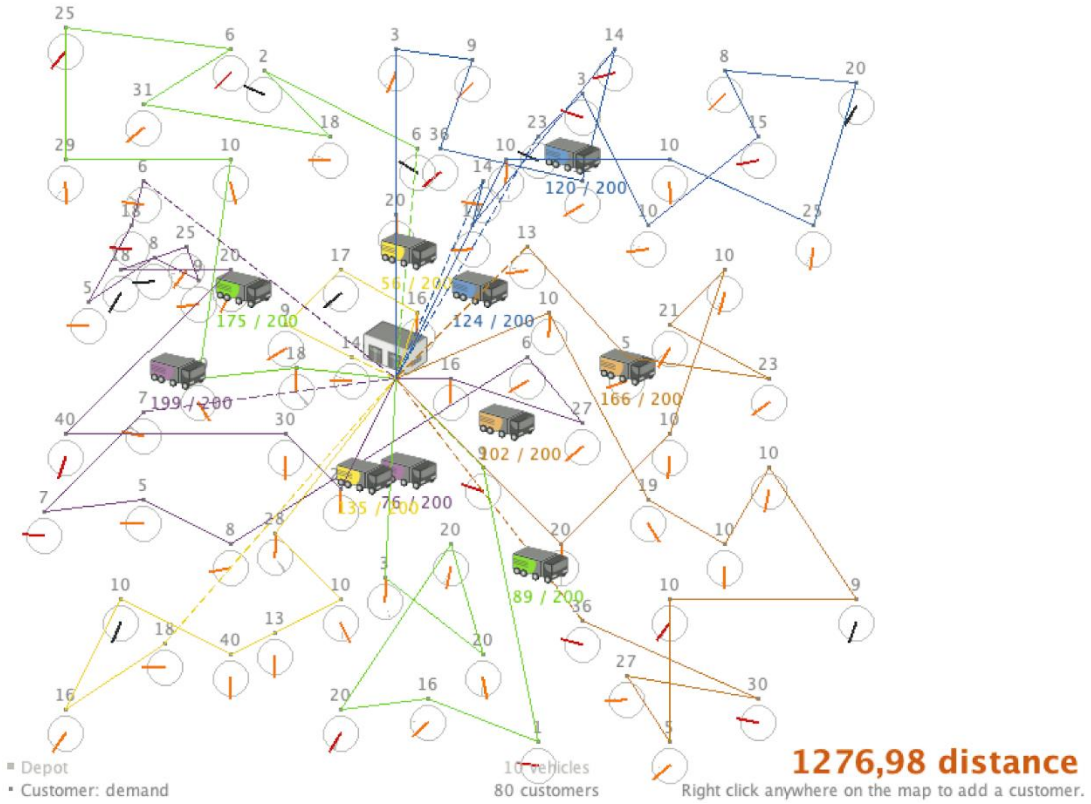

Figure 23: Résultat des tournées à 80 missions et 10 véhicules sans contrainte hard

OptaPlanner termine ses recherches de tournées optimales sur ce score : 1276,98. Le score est calculé en fonction des contraintes “soft” respectées et des objectifs d'optimisation du

problème. Les contraintes sont incluses dans la fonction objectif et correspondent aux fenêtres de temps des clients, des capacités des véhicules et des distances parcourues, tandis que les objectifs d'optimisation visent à minimiser la distance totale parcourue ou le nombre de véhicules utilisés :

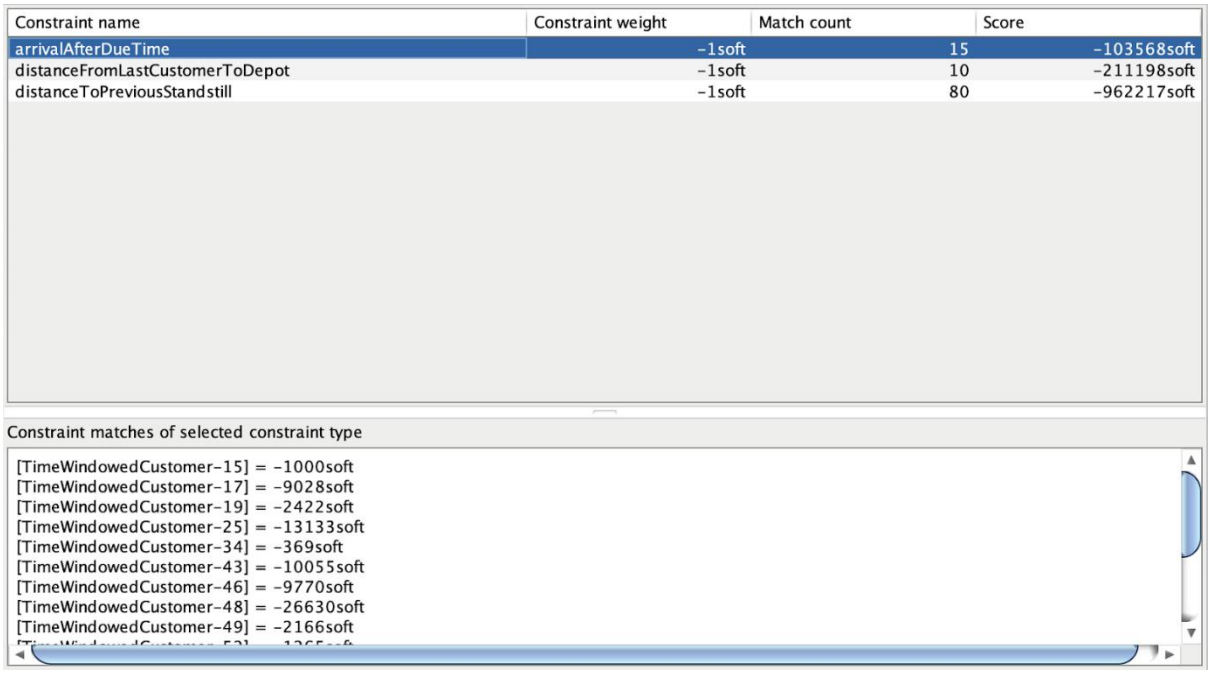

Figure 23: Aperçu des contraintes non respectées

Nous pouvons voir qu'en additionnant le score des 3 contraintes, nous obtenons 1 276 983. Divisé par 1000, on retrouve le score obtenu par OptaPlanner, 1276,58 correspond donc au score de la tournée optimale, d'après les calculs de l'outil. Cette tournée est représentée visuellement et montre le chemin que doit parcourir chaque véhicule pour atteindre l'optimisation trouvée par OptaPlanner. Plus le score est bas, plus la tournée est performante.

Par la suite, nous avons testé plusieurs jeux de données en variant le nombre de véhicules.

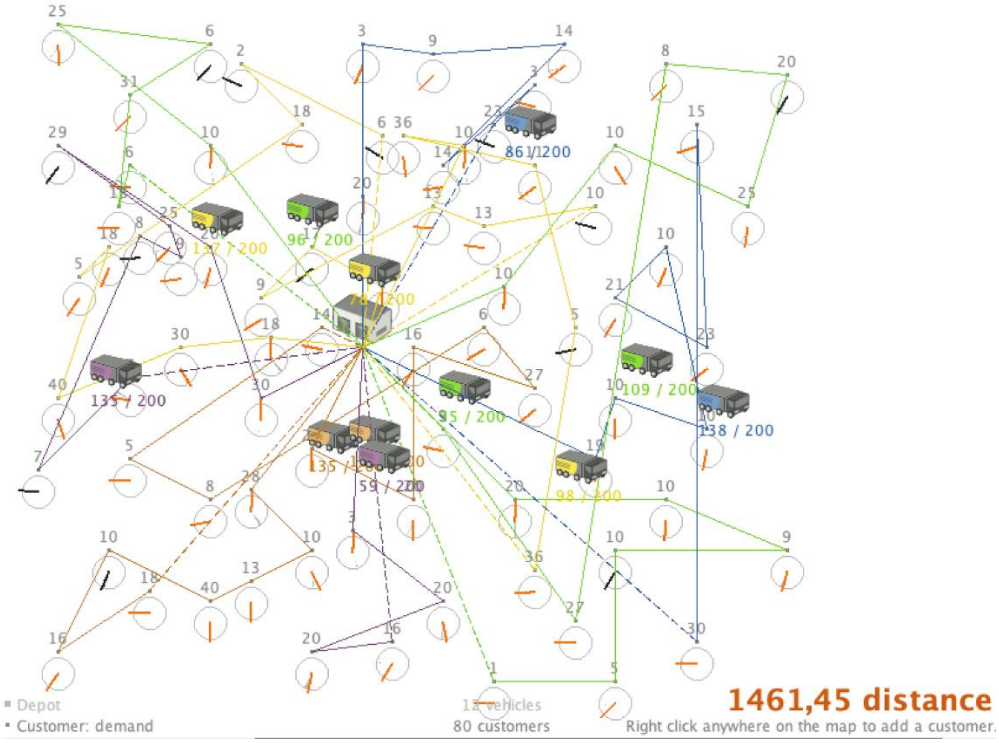

Figure 24: Résultat des tournées à 80 missions et 12 véhicules

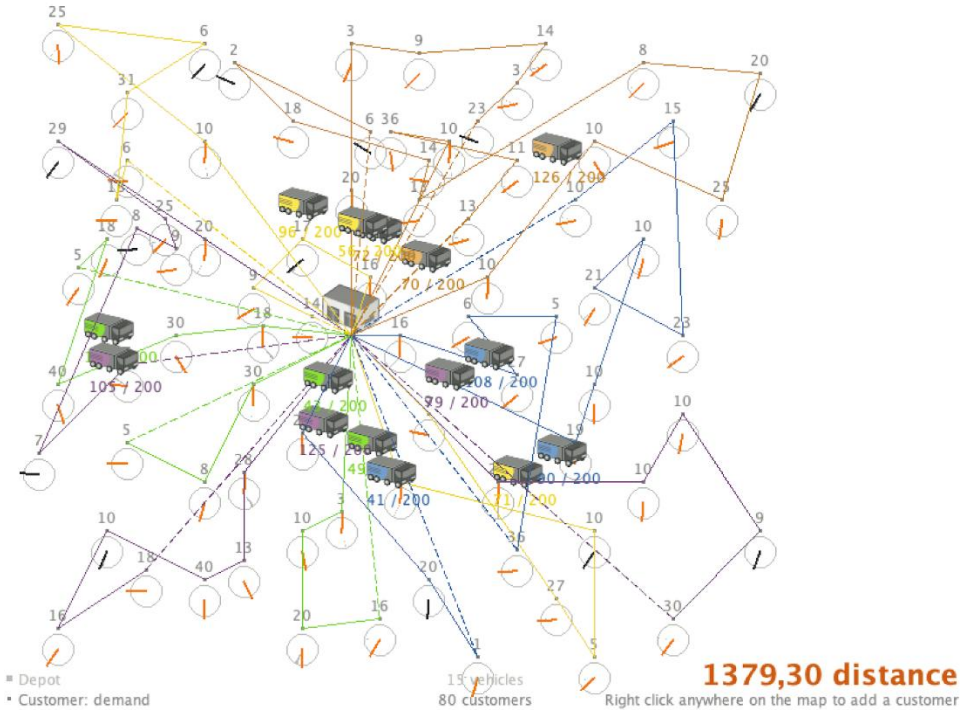

Figure 25: Résultat des tournées à 80 missions et 15 véhicules

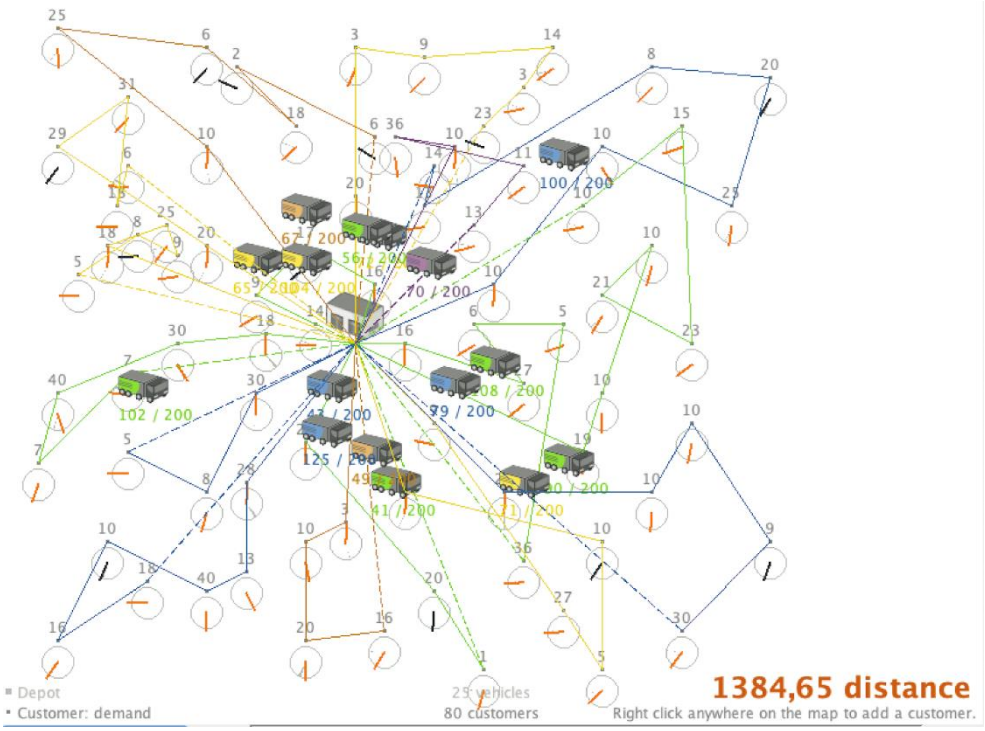

Figure 26: Résultat des tournées à 80 missions et 25 véhicules

Chacune des tournées les plus optimales sont affichées visuellement. Nous pouvons comparer nos résultats dans un tableau :

Type de tournée	Score
80 missions 10 véhicules	1276,95
80 missions 12 véhicules	1461,45
80 missions 15 véhicules	1379,30
80 missions 25 véhicules	1384,65

Tableau 5 : Différence de score entre les configurations

La différence de score entre les configurations est due à plusieurs facteurs.

Tout d'abord, avec un nombre plus élevé de véhicules, il y a une plus grande flexibilité dans la répartition des missions. Cela signifie qu'il est possible d'affecter les missions de manière plus équilibrée entre les véhicules, évitant ainsi des situations où certains véhicules sont surchargés tandis que d'autres sont sous-utilisés. Cette répartition plus équilibrée peut conduire à de meilleures performances en termes de respect de contraintes comme la fenêtre de temps des missions.

De plus, un plus grand nombre de véhicules peut également permettre de réduire les distances totales parcourues. En répartissant les missions entre un plus grand nombre de véhicules, il est possible d'optimiser les trajets et de minimiser les distances parcourues par chaque véhicule individuellement.

Concrètement, un plus grand nombre de véhicule permet une meilleure répartition des missions et le respect des contraintes de temps, afin d'améliorer le score global.

Cependant, nous remarquons que la tournée à 10 véhicules obtient le meilleur score, comparé à des tournées avec plus de véhicules. Cela peut être due au fait que nous avons transformé la contrainte "arrivalAfterDueTime" en "soft", autrement cette tournée n'est pas faisable par OptaPlanner.

Nous pouvons donc conclure au vu des résultats, selon OptaPlanner, que pour obtenir une tournée de 80 missions de façon optimisée, il faudrait 15 brancardiers. La distribution des missions pour chaque brancardier a été montrée précédemment.

2) Comparaison avec la méthode de programmation linéaire

En nous basant sur le même modèle que les indicateurs précédemment utilisés pour la programmation linéaire, nous avons :

Indicateurs retard

Nombre de brancardier	25	15	12	10
Retard cumulé	0	0	0	7

Tableau 6: indicateurs de retard rninutes

Seul le jeu de données avec 10 brancardiers admet un retard, en d'autres termes il n'a pas respecté la contrainte « hard » de fenêtre de temps en disposant de plusieurs brancardiers ayant un retard de 7 minutes au total.

Indicateur sur la charge de travail par brancardier en nombre de mission

Nombre de brancardier	25	15	12	10
Moyenne charge	3	5	6,5	8
Charge maximale	5	7	8	10
Charge minimale	2	3	6	6
Ecart moyen	1	1	0,6	1

Tableau 7: Indicateur sur la charge de travail par brancardier en nombre de mission

On peut observer un écart moyen très bas entre les missions affectées aux brancardiers. La charge maximale et minimale diminue logiquement en fonction du nombre de brancardiers.

Notre point de comparaison entre les résultats obtenus avec la programmation linéaire est le jeu de données avec 80 missions et 12 brancardiers :

Comparaison des indicateurs selon la méthode utilisée

Méthode	Programmation linéaire	OptaPlanner
Retard cumulé	0	0
Moyenne charge	8,6	6,5
Charge maximale	11	8
Charge minimale	5	6
Ecart moyen	1,6528	0,6

Tableau 8: Comparaison des indicateurs selon la méthode utilisée

Il y a une différence significative entre les résultats des deux méthodes utilisées. OptaPlanner résout des problèmes de CVRPTW, aucune mission ne nécessite 2 brancardiers, ce qui réduit drastiquement le nombre de missions par brancardier. En effet, la notion de charge dans OptaPlanner n'implique pas qu'une mission nécessite 2 brancardiers, ce qui impacte le résultat final comparé à celui de la programmation linéaire.

Cependant, l'écart moyen du nombre de missions réparties entre les brancardiers est plus bas sur OptaPlanner, bien que les caractéristiques techniques de l'outil soient différentes. On remarque une homogénéisation notable du nombre de missions entre brancardiers.

3. Comparaison des trois modèles

La méthode de programmation linéaire, la méta-heuristique et la heuristique sont toutes des approches utilisées pour résoudre des problèmes d'optimisation, mais elles diffèrent dans leur leurs caractéristiques.

Nous avons comparé la méthode basée sur la programmation linéaire et la méta-heuristique sur OptaPlanner, nous devons maintenant évaluer la méthode heuristique et ses performances par rapport aux deux méthodes précédentes.

Les simulations avec cette approche ont été réalisées par le groupe d'étudiant ayant travaillé sur ce projet avant nous. Le travail a été fait sur Python, à l'aide du jeu de données fourni par AMC, qui a aussi été utilisé par notre groupe pour les méthodes de programmation linéaire et méta-heuristique.

Les résultats pour la méthode heuristique sont les suivants :

Nombre de mission	1B	2B	Nombre de brancardier	Temps d'exécution (seconde)	Répartition (nb de missions)	Retard cumulé (seconde)
50	35	15	4	0.07	17,13,15,13	24000
50	35	15	5	0.12	17,11,15, 5, 9	16800
50	35	15	6	0.11	11,13,10, 7, 9,11	12600
80	57	23	4	0.14	30,24,21,23	51900
80	57	23	5	0.13	28,19,23,15,11	38400
80	57	23	6	0.12	22,20,13,18,13, 8	35700
80	57	23	7	0.13	21,11,20,13,11,10,13	25200
80	57	23	8	0.15	21,11,20,13,11,10,13,4	25200

Figure 27: Résultat avec la méthode heuristique

Nous allons nous intéresser uniquement aux simulations avec 80 missions, nous pouvons relever plusieurs points :

- Le temps d'exécution : extrêmement bas, il n'augmente que de 0,01 entre les missions à 4 brancardiers et à 8
- La répartition des missions : elle devient de moins en moins homogène au fur et à mesure de l'augmentation du nombre de brancardiers, allant jusqu'à avoir une différence de 17 missions entre deux brancardiers
- Le retard cumulé : un très grand nombre, qui diminue lorsqu'on augmente le nombre de brancardier

Malgré un temps d'exécution très bas, la méthode heuristique reflète un grand nombre de problèmes.

Tout d'abord, la répartition des missions n'est pas homogène comparé aux deux méthodes étudiées précédemment. En effet, pour une tournée à 80 missions et 8 brancardiers, il y a un écart de 17 missions effectuées entre le brancardier ayant effectué le maximum de mission et celui ayant fait le moins, alors que pour la méthode méta-heuristique par exemple l'écart maximum était de 4 sur une tournée à 10 brancardiers.

De plus, le retard cumulé diminue certes quand on augmente le nombre de brancardier, mais reste beaucoup plus grand que sur les deux autres méthodes. Il y a plus de 7h de retard

cumulé sur la tournée à 8 brancardiers, alors que sur CPLEX le plus grand retard obtenu est de 6 minutes.

Enfin, la simulation s'arrête à la tournée de 8 brancardiers, contrairement aux autres méthodes où nous pouvons monter jusqu'à 12, 15 voir 25 brancardiers. OptaPlanner ne peut pas descendre en dessous de 12 brancardiers pour 80 missions car cela induit qu'il y aurait des retards et donc jugé infaisable par l'outil. De même, sur CPLEX, on ne peut pas descendre en dessous de 9 brancardiers sous peine de voir le modèle jugé infaisable. Cette différence peut aussi être un bon point pour la méthode heuristique, car elle permet de simuler des tournées à moins de 9 brancardiers, contrairement aux deux méthodes.

VIII. Axes d'amélioration

1. Optaplanner

Pour améliorer notre problème de tournée de brancardiers sur OptaPlanner, nous avons identifié plusieurs axes d'amélioration.

Tout d'abord, il est essentiel de coder le reste des contraintes spécifiques à notre problématique et de les adapter à l'outil. Par exemple, nous devons mettre en place une contrainte "hard" permettant de garantir que les missions nécessitant deux brancardiers soient respectées, en tenant compte de l'assignation des brancardiers.

De plus, nous pouvons implémenter du code supplémentaire pour adapter OptaPlanner à notre problématique spécifique. Cela pourrait inclure l'ajout de la vitesse moyenne en tant que variable aux véhicules, ce qui aurait un impact sur le calcul des distances et des temps de trajet. Nous pourrions créer une classe dédiée au calcul des distances entre les localisations sans passer par des coordonnées spatiales, ce qui serait particulièrement utile compte tenu de l'absence de coordonnées dans nos jeux de données actuels.

Un autre axe d'amélioration consiste à adapter les jeux de données en intégrant des coordonnées spatiales réalistes correspondant à notre environnement réel. Cela permettrait une modélisation plus précise des distances et des temps de trajet, renforçant ainsi la pertinence de nos résultats.

Enfin, il est important de questionner l'utilisation même d'OptaPlanner pour résoudre notre problématique. Étant donné que nous avons une problématique de VRP classique, et que l'outil est principalement conçu pour des problèmes de CVRP et CVRPTW, il convient de se demander si notre problème peut réellement être traité sur cet outil. Il peut aussi être pertinent d'explorer d'autres approches, telles que le traitement du problème comme un order picking ou un job scheduling, qui pourraient mieux correspondre à notre problématique. Ces deux approches ont été suggérées par M. Pingaud et M. De Smet, le créateur d'OptaPlanner, lors d'un échange sur un forum concernant notre problématique.

2. Cplex

Le modèle actuel basé sur CPLEX présente une opportunité d'amélioration significative pour optimiser la charge de travail des brancardiers. Une analyse approfondie révèle que les brancardiers ne sont pas exploités à leur plein potentiel. En effet, en considérant une planification avec 12 brancardiers, ceux-ci ont en moyenne 600 minutes de temps de travail disponible, mais seulement 180 minutes en moyenne sont effectivement utilisées. Cette sous-utilisation des ressources conduit à une inefficacité et à une capacité inutilisée.

Même avec une réduction à 9 brancardiers, la situation demeure similaire. Les brancardiers ont en moyenne 600 minutes de temps de travail disponibles, mais seulement 237 minutes en moyenne sont utilisées, laissant une marge considérablement élevée d'inactivité.

Pour essayer d'améliorer ce modèle, il serait essentiel de repenser l'utilisation de leur temps de travail. Cela pourrait inclure une meilleure répartition des missions, une optimisation des trajets ou l'introduction de mécanismes pour combler les périodes d'inactivité. En exploitant plus efficacement le temps de travail des brancardiers, il serait possible d'augmenter la productivité et d'obtenir une utilisation plus optimale des ressources disponibles.

IX. Gestion du projet et de l'équipe

Dans le cadre de notre projet tutoré, nous avons opté pour l'utilisation de la méthode agile pour la gestion de projet. Cette approche nous a permis de gérer efficacement les tâches et d'assurer une collaboration dynamique au sein de notre équipe. La répartition des tâches a été réalisée de manière collaborative, en tenant compte des compétences et des préférences de chaque membre.

Pour faciliter la coordination et la centralisation des tâches, nous avons utilisé l'outil Trello. Trello est une plateforme en ligne qui facilite la gestion et l'organisation des tâches, des projets et des équipes. Avec Trello, les utilisateurs peuvent créer des tableaux virtuels où ils peuvent organiser leurs tâches sous forme de listes et de cartes. Chaque carte représente une tâche ou une activité spécifique, et peut être déplacée entre différentes listes pour refléter son statut (à faire, en cours, terminée, etc.).

Cela nous a permis de créer des tableaux de bord pour organiser les tâches, les ressources et définir les délais.

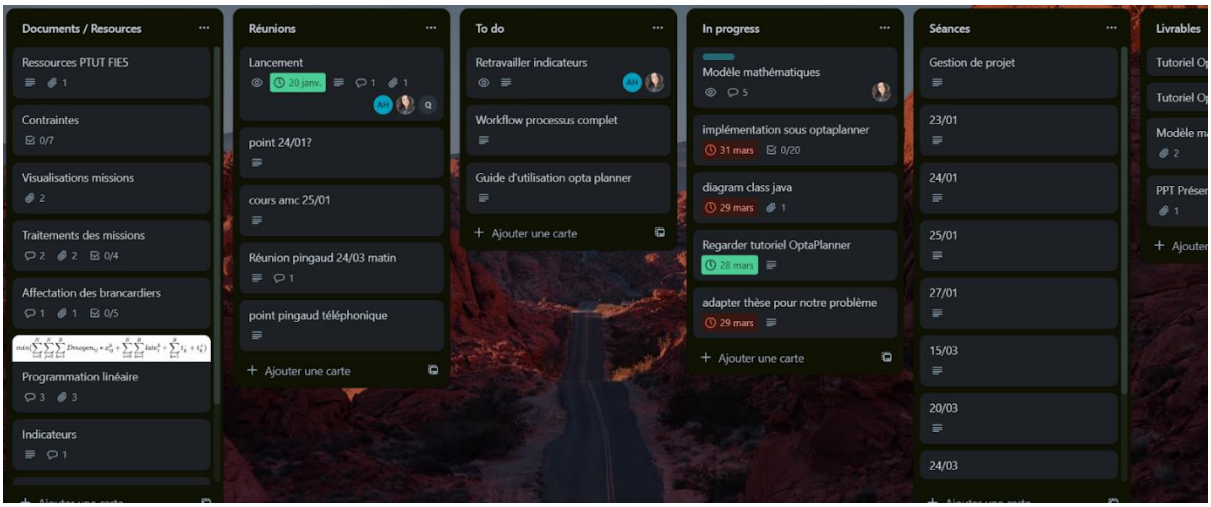

Figure 29: Aperçu de notre tableau Trello

Nous avons régulièrement tenu des points de suivi pour s'organiser et nous tenir mutuellement informés des avancées. Ces réunions nous ont également donné l'occasion d'ajuster les directions prises et de prendre des décisions en fonction de l'évolution du projet. Grâce à cette approche agile, nous avons pu maintenir une flexibilité nécessaire pour répondre aux changements et aux défis rencontrés tout au long du projet, tout en garantissant une gestion efficace des ressources et des délais.

X. Conclusions et perspectives

Dans le cadre de ce projet, notre objectif principal était de comparer différentes méthodes d'optimisation des flux de brancardiers. L'objectif final était de recommander la solution la plus adaptée aux besoins de l'entreprise et de ses clients. Pour atteindre cet objectif, nous avons étudié trois méthodes différentes : la méthode heuristique déjà utilisée en interne, une méthode basée sur la programmation linéaire exacte, et enfin une méthode méta-heuristique utilisant l'outil OptaPlanner.

La méthode heuristique présente certaines limites en ce qui concerne la minimisation des retards, ainsi que lorsqu'elle est confrontée à des ensembles de données de grande taille. Bien que cette approche puisse fournir des solutions rapides et acceptables dans de nombreux cas, elle peut avoir du mal à trouver des solutions optimales lorsqu'il y a un grand nombre de contraintes et de variables à prendre en compte. De plus, la méthode heuristique peut être moins précise dans la gestion des retards, ce qui peut entraîner des performances suboptimales en termes de satisfaction des contraintes de temps. Par conséquent, il est important d'explorer d'autres approches, telles que la méthode exacte en programmation linéaire, pour surmonter ces limites et obtenir des résultats plus précis et optimisés.

Nous avons exploré l'utilisation d'OptaPlanner, un moteur de planification de contraintes. Nous avons importé les données de notre problème dans un format adapté à OptaPlanner. Cependant, nous avons rencontré certaines problématiques spécifiques.

Au dépit de nos efforts pour utiliser OptaPlanner dans notre projet d'optimisation des tournées de brancardier, nous avons finalement rencontré des difficultés qui ont abouti à un échec. OptaPlanner, bien qu'étant un outil puissant pour la résolution de problèmes de planification avec des contraintes, n'était pas adapté à notre problématique spécifique. Nous avons constaté que les fonctionnalités et les modèles proposés par OptaPlanner étaient plus adaptés aux CVRP, tandis que notre problème de tournées de brancardiers présentait des caractéristiques différentes et des contraintes plus complexes.

Malgré ces problématiques, nous avons identifié plusieurs axes d'amélioration potentiels pour notre sujet. Il serait pertinent de remettre en question l'utilisation d'OptaPlanner pour notre problématique, en considérant si notre problème de tournées de brancardiers peut être réellement traité comme un VRP ou s'il serait plus adapté d'utiliser d'autres approches telles que l'order picking ou le job scheduling. Par exemple, on peut minimiser la distance totale parcourue par les brancardiers lors de l'exécution des missions dans un problème d'order picking. De même, si les missions doivent être planifiées dans un certain ordre en tenant compte des contraintes de temps et des ressources disponibles, la méthode du job scheduling peut être utilisée.

Enfin nous avons poussé au maximum la méthode exacte avec la programmation linéaire. Grâce à son utilisation, nous avons obtenu des résultats remarquables dans l'optimisation des tournées des brancardiers. Cette approche rigoureuse a permis de modéliser et de résoudre de manière précise et efficace le problème complexe de planification des brancardiers. En prenant en compte les multiples contraintes et objectifs, tels que les horaires de départ, les temps de trajet, les capacités de travail et les contraintes de successeurs, nous avons pu générer des tournées optimales pour chaque brancardier.

En conclusion, la méthode exacte en programmation linéaire se révèle être une solution prometteuse pour l'optimisation des tournées des brancardiers. Malgré sa complexité et sa demande en ressources de calcul, cette approche offre des résultats plus précis et optimisés par rapport à la méthode heuristique. Elle permet de prendre en compte de manière rigoureuse toutes les contraintes et objectifs du problème, notamment la minimisation des retards, la

satisfaction des contraintes de temps et la gestion efficace des grandes quantités de données. En choisissant la méthode exacte, nous nous assurons d'obtenir une solution optimale et de qualité, garantissant ainsi une meilleure efficacité des opérations de transport des brancardiers. Toutefois, il est important de noter que l'utilisation de cette méthode peut nécessiter des ressources computationnelles importantes, en particulier pour les jeux de données volumineux. Il convient donc de prendre en compte ces aspects lors de la sélection de l'approche d'optimisation la plus adaptée.

Pour les suites à donner au projet, nous recommandons de poursuivre le travail sur OptaPlanner. L'objectif serait d'approfondir les possibilités d'amélioration de l'utilisation maximale de la capacité de travail des brancardiers en utilisant la méthode exacte en programmation linéaire. Cette approche offre un potentiel prometteur pour optimiser davantage l'affectation des brancardiers et maximiser leur productivité. En affinant le modèle et en ajustant les contraintes et les objectifs, il serait possible de mieux exploiter les ressources disponibles et d'obtenir des résultats encore plus efficaces et précis. Cela permettrait de mieux répondre aux besoins opérationnels de l'entreprise et d'améliorer encore davantage la qualité des tournées des brancardiers. Il conviendrait également d'évaluer l'impact de ces améliorations sur les performances de calcul, en tenant compte des ressources informatiques disponibles. En résumé, en poursuivant les efforts sur OptaPlanner et en explorant les opportunités offertes par la méthode exacte, il est possible de tirer parti de leur complémentarité pour atteindre des niveaux d'optimisation plus élevés dans la gestion des tournées des brancardiers

XI. Bibliographie

[1] Bouabdallah, M.N., Rached, M., Fondrevelle, J., Bahroun, Z., (2013). “Organization and management of hospital patient transportation system”.

[2] Malapert, A. (2006). “Optimisation de tournées de Véhicules pour l’exploitation de Réseau Telecom”.

[3] Zhang L. (). “De la vision métier à la génération assistée de plannings pour la coordination centralisée de services de soins à domicile”.

[4] Ducomman S. (2018). “Optimisation de tournées de véhicules par programmation par contraintes : conception et développement d’un solveur industriel”.

[5] Maintenant V. (2022). “Programmation Linéaire et Optimisation du brancardage”.

[6] Rogers N. (2022). “Optimisation du flux de brancardiers”.

[7] Coubetergues A, De Quina D. (2022). “Projet régulation brancardages”.

[8] AMC (2022). “Compte-rendu Sarreguemines Brancardier”.

[9] Ducomman S. (2018). “Optimisation de tournées de véhicules par programmation par contraintes : conception et développement d’un solveur industriel”.

[10] Dr. Kherici N. (2020). “Méthode de résolution en optimisation combinatoire”.

Annexe

Modèle mathématique

CF fichier pdf dans le dossier des annexes nommé : “Annexe_ModeleMathematique.pdf”

Modèle CPLEX

CF fichier .mod dans le dossier des annexes nommé : “modele.mod”

Data pour CPLEX

CF fichier .dat dans le dossier des annexes nommé : “data.dat”

Data pour OptaPlanner

CF fichiers .json dans le dossier des annexes nommés : “cvrptw-brancardier.json”

Table des Figures

Table des équations

Table des tableaux

Glossaire

OptaPlanner : Un outil open-source d'optimisation de contraintes basé sur des algorithmes de satisfaction de contraintes. Il est utilisé pour résoudre des problèmes d'ordonnancement, de planification et d'optimisation dans divers domaines.

CPLEX : Un logiciel d'optimisation mathématique commercial développé par IBM. Il offre des fonctionnalités avancées pour résoudre des problèmes d'optimisation linéaire, mixte, quadratique et non linéaire.

Figma : Une plateforme de conception d'interfaces utilisateur (UI) et de prototypage collaboratif basée sur le cloud. Elle permet aux équipes de concevoir, de prototyper et de collaborer sur des projets de conception d'interfaces utilisateur.

Heuristique : Une méthode de résolution de problèmes qui repose sur des règles empiriques, des approximations et des techniques basées sur l'expérience plutôt que sur des calculs mathématiques rigoureux. Les heuristiques fournissent des solutions réalisables mais ne garantissent pas l'optimalité.

Métaheuristique : Une approche d'optimisation qui utilise des stratégies générales pour explorer l'espace des solutions d'un problème. Les métaheuristiques sont utilisées pour résoudre des problèmes d'optimisation complexes et peuvent être appliquées à différentes classes de problèmes sans être spécifiquement adaptées à chacun.

Méthode exacte : Une méthode d'optimisation qui garantit la recherche de la solution optimale pour un problème donné en utilisant des techniques mathématiques rigoureuses. Elle repose sur la modélisation mathématique du problème et l'utilisation d'algorithmes de résolution précis pour trouver la solution optimale, mais peut être limitée par la taille et la complexité du problème.

GHT : Groupements Hospitaliers de Territoire

CVRP (Capacitated Vehicle Routing Problem) : Problème de routage de véhicules avec capacité. Il s'agit d'un problème d'optimisation dans lequel des véhicules doivent être affectés à un ensemble de clients tout en respectant les contraintes de capacité des véhicules.

VRPTW (Vehicle Routing Problem with Time Windows) : Problème de routage de véhicules avec fenêtres de temps. Il s'agit d'un problème d'optimisation qui inclut des contraintes de temps spécifiques pour les visites des clients. Les véhicules doivent respecter les fenêtres de temps prédéfinies pour effectuer les livraisons ou les collectes.

PVRP (Period Vehicle Routing Problem) : Problème périodique de routage de véhicules. Il s'agit d'un problème d'optimisation où les véhicules doivent être planifiés sur plusieurs périodes de temps, en tenant compte des contraintes de capacité, des fenêtres de temps et des demandes des clients.

IRP (Inventory Routing Problem) : Problème de routage des stocks. Il s'agit d'un problème d'optimisation dans lequel les livraisons doivent être planifiées en tenant compte des niveaux de stock des clients. Les véhicules doivent effectuer des livraisons tout en reconstituant les stocks de manière efficace.

M-VRPTW (M-Vehicle Routing Problem with Time Windows) : Problème de routage de plusieurs véhicules avec fenêtres de temps. Il s'agit d'une extension du VRPTW dans laquelle plusieurs véhicules doivent être planifiés pour desservir un ensemble de clients en respectant les contraintes de capacité et les fenêtres de temps.

VRP (Vehicle Routing Problem) : Problème de routage de véhicules. Il s'agit d'un problème d'optimisation où des véhicules doivent être affectés à un ensemble de clients afin d'optimiser les itinéraires, en prenant en compte les contraintes de capacité, les distances à parcourir et éventuellement les fenêtres de temps.

Trello : Trello est une plateforme en ligne qui facilite la gestion et l'organisation des tâches, des projets et des équipes.

Méthode Agile : Une approche de gestion de projet basée sur l'itération et la collaboration. Les méthodes agiles favorisent la flexibilité, l'adaptabilité et la livraison continue de produits ou de résultats de haute qualité. Elles mettent l'accent sur la communication régulière avec les parties prenantes, l'auto-organisation des équipes et l'ajustement constant des objectifs en fonction des retours d'expérience.