



ÉCOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE ET D'ANALYSE DES SYSTÈMES -
ENSIAS

Filière : SMART SUPPLY CHAIN AND LOGISTICS

Rapport du projet :
ATELIER DE MODELISATION

Auteurs :

WAHI Yacine

EL MADHOUN Kaouthar

Encadré par :

MONSIEUR. EL AMRANI MOHAMMED

MONSIEUR. ABBAL KHALIL

Année Universitaire 2023-2024



Remerciements :

Avant d'entamer la rédaction de ce rapport, nous exprimons notre profonde gratitude envers toutes les personnes ayant contribué, de manière directe ou indirecte, au succès de ce projet d'atelier de modélisation. Nous tenons à remercier chaleureusement nos encadrants **Monsieur EL Amrani Mohammed**, et **Monsieur Abbal Khalil**, pour leur accompagnement continu, depuis la modélisation initiale du problème jusqu'à la réalisation concrète du projet.



Résumé :

Ce rapport expose la conception et la résolution d'un problème combinant deux défis majeurs de l'optimisation logistique : la localisation multi-capacité avec contrainte budgétaire et la localisation avec plusieurs niveaux d'affectation et contraintes de capacité. L'objectif est de modéliser de manière réaliste la prise en compte de plusieurs niveaux d'affectation et de capacité pour chaque installation. Le projet utilise Java avec CPLEX pour la modélisation et la résolution, tandis que Python est employé pour la génération d'instances. La première partie du rapport contextualise la logistique, détaille les deux problèmes étudiés, et propose une modélisation mathématique avancée combinant ces deux problèmes avec des contraintes spécifiques. La seconde partie traite de la résolution en mettant en œuvre des outils tels que JavaCPLEX et Python. En conclusion, le rapport présente le prototype final, démontrant la concrétisation pratique des concepts abordés. Cette approche offre une compréhension approfondie des problématiques logistiques complexes, soulignant le rôle crucial de l'optimisation linéaire dans des prises de décision efficaces et la résolution de problèmes pratiques.

Abstract :

This report outlines the design and resolution of a problem combining two major challenges in logistic optimization : multi-capacity location with budget constraints and location with multiple assignment levels and capacity constraints. The goal is to model realistically the consideration of multiple assignment levels and capacity for each facility. The project utilizes Java with CPLEX for modeling and solving, while Python is employed for instance generation. The first part of the report contextualizes logistics, details the two studied problems, and proposes advanced mathematical modeling that combines these two problems with specific constraints. The second part addresses the resolution by implementing tools such as JavaCPLEX and Python. In conclusion, the report presents the final prototype, demonstrating the practical realization of the discussed concepts. This approach provides an in-depth understanding of complex logistic challenges, emphasizing the crucial role of linear optimization in effective decision-making and problem-solving in practical scenarios.

Table des matières

| | |
|--|-----------|
| Table des figures | 5 |
| Introduction générale | 6 |
| 1 Présentation générale du projet | 7 |
| 1.1 Contexte : | 7 |
| 1.1.1 Problème 1 : | 7 |
| 1.1.2 Problème 2 : | 8 |
| 1.2 Définition du problème résultant | 8 |
| 1.3 Exemple illustrateur : | 9 |
| 2 MODELISATION MATHEMATIQUE | 11 |
| 2.1 Modélisation du problème de multi-capacités : | 11 |
| 2.1.1 Les paramètre et les variables | 11 |
| 2.1.2 Le modèle mathématique du problème : | 11 |
| 2.1.3 Explication des contraintes : | 12 |
| 2.2 Modélisation du problème multi-niveaux : | 12 |
| 2.2.1 Les paramètre et les variables : | 12 |
| 2.2.2 Le modèle mathématique du problème : | 13 |
| 2.2.3 Explication des contraintes : | 13 |
| 2.3 Modélisation du problème résultant : | 14 |
| 2.3.1 Les paramètre et les variables : | 14 |
| 2.3.2 le modèle mathématique du problème résultant | 14 |
| 2.3.3 Explication des contraintes : | 15 |
| 3 RESOLUTION | 16 |
| 3.1 Techniques et outils utilise | 17 |
| 3.1.1 Java et CPLEX | 17 |
| 3.1.2 Python | 18 |
| 3.2 Réalisation : | 19 |
| 3.2.1 Résolution par les instances du 1er test : | 19 |
| 3.2.2 Résolution par les instances du générateur : | 19 |
| Conclusion générale et perspectives | 21 |

Table des figures

| | | |
|-----|---|----|
| 1.1 | Schéma illustateur du problème. | 8 |
| 3.1 | Schema illustratif du résolution | 16 |
| 3.2 | IBM-CPLEX-Logo | 17 |
| 3.3 | Java-Logo | 18 |
| 3.4 | Python-Logo | 18 |
| 3.5 | Résultats du 1er test avec B=10000. | 19 |
| 3.6 | Résultats par les instances du générateur | 19 |

Introduction générale :

L'optimisation logistique, au cœur des préoccupations de la gestion des chaînes d'approvisionnement et de la planification opérationnelle, constitue un enjeu majeur dans la quête incessante d'efficacité et de réduction des coûts par les entreprises. Dans ce contexte, l'application de techniques d'optimisation linéaire s'avère cruciale. Ce projet de modélisation -comme il est déjà mentionné- se concentre sur deux problématiques fondamentales : la localisation multi-capacité avec contrainte budgétaire et la localisation avec plusieurs niveaux d'affectation et contraintes de capacité.

L'objectif principal réside dans le développement de modèles mathématiques avancés, exploitant notamment l'optimisation linéaire, pour représenter de manière réaliste les défis complexes inhérents à ces problèmes logistiques. La résolution de ces modèles s'effectuera à l'aide d'outils puissants tels que Java avec CPLEX pour la modélisation et la résolution, ainsi que Python pour la génération d'instances. Cette approche intégrée démontre comment l'optimisation linéaire peut jouer un rôle crucial dans la prise de décisions efficaces et la résolution de problèmes pratiques liés à la logistique.

Dans cette perspective, le projet d'atelier de modélisation se structure en plusieurs sections, débutant par une introduction générale du projet et des problèmes étudiés. Nous approfondirons ensuite les modèles mathématiques élaborés pour chaque problématique, mettant en exergue les paramètres, variables et contraintes spécifiques. La résolution de ces modèles sera abordée en détail, illustrant l'utilisation de Java avec CPLEX et Python, avec une analyse approfondie des résultats obtenus.

Chapitre 1

Présentation générale du projet

1.1 Contexte :

Le domaine de la logistique est confronté à des défis croissants en matière d'optimisation des ressources et de prise de décisions efficaces. Dans ce contexte, les problèmes d'optimisation linéaire jouent un rôle crucial pour améliorer l'efficacité opérationnelle, réduire les coûts et optimiser les processus logistiques. Par exemple, les entreprises de transport doivent planifier efficacement l'affectation des ressources, telles que les véhicules et les entrepôts, pour répondre à la demande des clients tout en minimisant les coûts d'exploitation. De même, les entreprises de fabrication doivent optimiser l'allocation des ressources de production pour maximiser l'efficacité et minimiser les temps d'arrêt. Dans ce contexte, notre projet vise à explorer deux problèmes d'optimisation distincts : la localisation multi-capacité avec contrainte budgétaire et la localisation avec plusieurs niveaux d'affectation et des contraintes de capacité. En combinant ces deux problèmes, notre objectif est de modéliser et résoudre un cas plus réaliste, tenant compte de la présence de plusieurs niveaux d'affectation et de capacité pour chaque installation. Cette approche permettra de mieux comprendre les défis pratiques auxquels sont confrontées les entreprises dans la gestion de leurs opérations logistiques, et d'offrir des solutions plus efficaces et adaptées à ces défis.

1.1.1 Problème 1 :

Le problème de localisation multi-capacité avec contrainte budgétaire est un défi d'optimisation logistique visant à déterminer la répartition optimale des clients parmi un ensemble d'installations, tout en considérant plusieurs niveaux de capacité pour chaque installation. L'objectif principal est de minimiser les coûts d'affectation des clients tout en respectant des contraintes strictes, notamment la capacité maximale de chaque installation et la restriction budgétaire pour l'ouverture de ces installations.

Ce problème trouve son application dans divers domaines, tels que la planification de réseaux logistiques, la localisation d'entrepôts ou de centres de distribution, où des contraintes budgétaires et des capacités opérationnelles doivent être prises en compte. Les décisions concernant l'affectation des clients à différentes installations et les niveaux d'ouverture de ces installations

ont un impact direct sur les coûts totaux et l'efficacité opérationnelle.

1.1.2 Problème 2 :

Le problème de localisation avec plusieurs niveaux d'affectation et contraintes de capacité est un défi d'optimisation logistique. Il consiste à déterminer stratégiquement les installations à ouvrir à différents niveaux, influençant ainsi le cheminement optimal des produits depuis les sources jusqu'aux clients finaux. L'objectif majeur est de minimiser les coûts totaux d'affectation tout en garantissant la satisfaction des demandes des clients. Les niveaux d'affectation intermédiaires ajoutent une complexité, nécessitant une optimisation fine pour l'équilibre des flux et la gestion des capacités.

Dans ce problème plusieurs contraintes sont à prendre en compte. Tout d'abord, il est essentiel de s'assurer que la demande des clients finaux est entièrement satisfaite. Ensuite, il faut respecter les capacités limitées des installations à chaque niveau pour éviter des contraintes opérationnelles excessives. De plus, maintenir un équilibre adéquat du flux de produits à travers les installations intermédiaires est crucial pour garantir une distribution efficace. Enfin, des contraintes d'intégrité doivent être imposées pour garantir la cohérence des décisions concernant l'ouverture des installations à chaque niveau.

1.2 Définition du problème résultant

Explication illustratrice de la problématique objectives :

Le problème combiné de localisation multi-capacité avec plusieurs niveaux d'affectation pose un défi d'optimisation logistique majeur. Il s'agit de déterminer de manière stratégique l'emplacement optimal des installations en tenant compte de plusieurs niveaux d'affectation et de capacités variées.

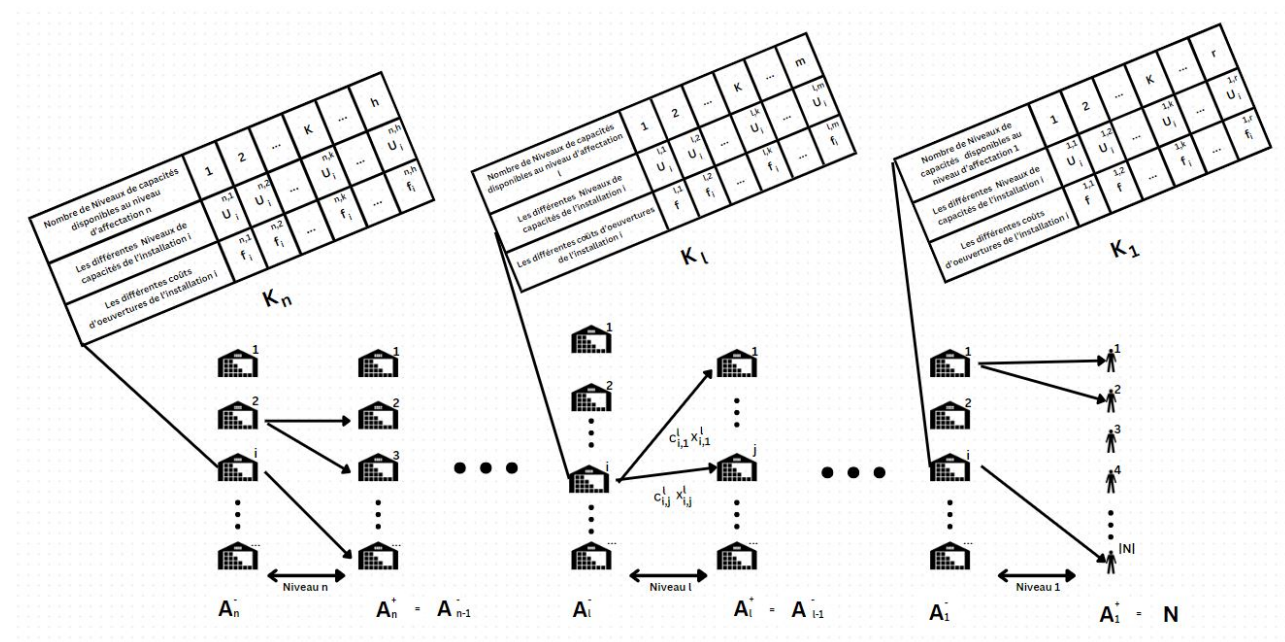


FIGURE 1.1 – Schéma illustrateur du problème.

Dans ce problème complexe, chaque niveau d'affectation, noté l , est caractérisé par un ensemble spécifique de niveaux de capacité pour les installations, dénoté K_l . Le nombre de niveaux de capacité, ou cardinal de K_l , dépend du niveau l . Chaque installation i dans un niveau d'affectation l a différents niveaux de capacité, notés $U_i^{l,k}$ et a un coût d'ouverture $f_i^{l,k}$, où k représente le niveau de capacité spécifique choisi parmi ceux disponibles dans K_l . Pour chaque niveau d'affectation l , les fournisseurs et les clients sont respectivement notés A_l^- et A_l^+ . La quantité de produits fournie de l'installation i à l'installation j au niveau l est symbolisée par X_{ij}^l , et le coût unitaire associé est C_{ij}^l . Ces caractéristiques détaillées enrichissent la modélisation du problème en tenant compte des nuances spécifiques à chaque niveau d'affectation et de capacité.

L'objectif principal est de minimiser les coûts tout en répondant à la demande des clients finaux. Les contraintes essentielles incluent la satisfaction des demandes clients, le respect des capacités opérationnelles des installations à chaque niveau, la sélection exclusive d'un niveau de capacité, le maintien de l'équilibre du flux de produits à travers les niveaux intermédiaires, la gestion rigoureuse des contraintes budgétaires, et l'assurance de l'intégrité des décisions d'ouverture des installations. Ce problème complexe requiert une modélisation mathématique avancée et l'utilisation d'outils d'optimisation pour obtenir des solutions optimales, tout en répondant efficacement aux exigences pratiques des réseaux logistiques.

1.3 Exemple illustrateur :

Prenons l'exemple d'une chaîne d'approvisionnement alimentaire pour illustrer le problème d'optimisation linéaire multi-capacité et multi-niveau. Imaginons plusieurs niveaux d'affectation, allant des usines de production aux épiceries de quartier.

Au niveau le plus bas (niveau 1), les épiceries de quartier sont les clients finaux (A_1^+). Elles s'approvisionnent exclusivement auprès des entrepôts du niveau 1 (A_1^-). Ces entrepôts, agissant comme fournisseurs pour les épiceries, reçoivent leurs approvisionnements de fournisseurs de niveau 2. Cette relation de fournisseur-client se poursuit à travers chaque niveau, avec chaque niveau agissant à la fois comme fournisseur pour le niveau inférieur et client pour le niveau supérieur. Cette structure hiérarchique persiste jusqu'aux usines de production, qui sont les fournisseurs de plus haut niveau.

L'optimisation linéaire multi-capacité et multi-niveau intervient ici pour minimiser les coûts tout en satisfaisant la demande des clients finaux. Chaque entrepôt a différentes capacités d'entreposage représentées par $U_i^{l,k}$, variant en fonction du niveau et du type d'entrepôt. Les épiceries, en tant que clients finaux, s'approvisionnent exclusivement à partir des entrepôts de leur niveau, mettant en lumière l'importance de l'efficacité dans la distribution de produits à chaque étape de la chaîne.

Ce problème d'optimisation vise à rationaliser la chaîne d'approvisionnement, minimisant

les coûts tout en assurant une distribution efficace. Il est crucial dans la gestion logistique, contribuant à réduire les dépenses opérationnelles et à garantir que les produits atteignent les clients finaux de manière efficace, en tenant compte des contraintes de capacité à chaque niveau. En résumé, cette optimisation linéaire joue un rôle essentiel dans la réalisation d'une chaîne d'approvisionnement efficace et économique.

Chapitre 2

MODELISATION MATHEMATIQUE

Dans ce chapitre, nous présentons les problèmes d'optimisation logistique mentionnés au chapitre 1 à travers des modèles mathématiques détaillés. Ces modèles constituent l'ossature analytique qui permettra la résolution précise et efficace des défis logistiques complexes rencontrés.

2.1 Modélisation du problème de multi-capacités :

2.1.1 Les paramètre et les variables

N : Ensemble de clients.

M : Ensemble d'installations.

L : Ensemble des niveaux de capacité.

d_i : Demande du client i , $i \in N$.

c_{ij} : Coût d'affectation du client i à l'installation j , $i \in N$, $j \in M$.

u_k : Capacité du niveau k .

f_k : Coût d'ouverture du niveau k exploité.

$$x_{ij} = \begin{cases} 1, & \text{si le client } i \text{ est affecté à l'installation } j \\ 0, & \text{sinon} \end{cases}$$

$$y_{jk} = \begin{cases} 1, & \text{si l'établissement } j \text{ est ouvert et utilisé au niveau } k \\ 0, & \text{sinon} \end{cases}$$

2.1.2 Le modèle mathématique du problème :

$$\text{Min} \sum_{i \in N} \sum_{j \in M} d_i * c_{ij} * x_{ij} \quad (1)$$

sujet a :

$$\sum_{j \in M} x_{ij} = 1 \quad \forall i \in N \quad (2)$$

$$\sum_{i \in N} d_i * x_{ij} \leq \sum_{k \in L} u_k * y_j^k \quad \forall j \in M \quad (3)$$

$$x_{ij} \leq \sum_{k \in L} y_j^k \quad \forall i \in N, \forall j \in M \quad (4)$$

$$\sum_{k \in L} y_{jk} \leq 1 \quad \forall j \in M \quad (5)$$

$$\sum_{j \in M} \sum_{k \in L} f^k * y_j^k \leq B \quad (6)$$

avec :

$$x_{ij} \in \{0, 1\} \quad \forall i \in N, \forall j \in M \quad (7)$$

$$y_{jk} \in \{0, 1\} \quad \forall j \in M, \forall k \in L \quad (8)$$

2.1.3 Explication des contraintes :

La fonction objective (1) consiste à minimiser les coûts d'affectation. Les contraintes (2) garantissent l'affectation de chaque client à une et une seule installation. Les contraintes (3) sont utilisées pour interdire l'affectation d'un client à une installation fermée. Les contraintes (4) sont des contraintes de capacité. Les contraintes (5) limitent l'ouverture du niveau de capacité à un seul niveau. La contrainte (6) est utilisée pour fixer une limite budgétaire aux coûts totaux d'ouverture. Enfin, les contraintes (7) et (8) sont des contraintes d'intégralité.

2.2 Modélisation du problème multi-niveaux :

2.2.1 Les paramètre et les variables :

L : Ensemble des niveaux d'affectation.

A^l : Ensemble des arcs (i, j) au niveau d'affectation l .

A^{l+} : Ensemble des installations j telles qu'il existe $i, (i, j) \in A^l$ (représentent les fournisseurs pour le niveau l).

A^{l-} : Ensemble des installations i telles qu'il existe $j, (i, j) \in A^l$ (représentent les clients pour le niveau l).

c_{ijl} : Coût d'affectation de l'installation i à l'installation j au niveau d'affectation l .

d_j : Demande du client j .

u_{il} : Capacité de l'installation i au niveau d'affectation l .

f_{il} : Coût d'ouverture de l'installation i au niveau d'affectation l .

$x_{ijl} \in R^+$: Quantité de produit fournie par l'installation i à l'installation j au niveau d'affectation l .

$$y_{il} \in \{0, 1\} : \begin{cases} 1, & \text{si l'installation } i \text{ au niveau } l \text{ est ouverte et utilisée} \\ 0, & \text{sinon} \end{cases}$$

2.2.2 Le modèle mathématique du problème :

$$\text{Min} \sum_{l=1}^{|L|} \left(\sum_{(i,j) \in A_l^2} c_{ij}^l x_{ij}^l \right) \quad (9)$$

Sujet a :

$$\sum_{i \in A_1^-} x_{ij}^1 = d_j \quad \forall j \in A_1^+ \quad (10)$$

$$\sum_{j \in A_l^+} x_{ij}^l \leq y_i^l * u_i^l \quad \forall l \in L, \forall i \in A^l - \quad (11)$$

$$\sum_{i \in A_l^-} x_{ij}^l \leq \sum_{p \in A_{l-1}^+} x_{jp}^{l-1} \quad \forall j \in A_l^+, \forall l \in L \setminus \{1\} \quad (12)$$

(2.1)

avec

$$x_{ij}^l \in R^+ \quad \forall l \in L, \forall (i, j) \in A_l^2 \quad (13)$$

$$y_i^l \in \{0, 1\} \quad \forall l \in L, \forall i \in K \quad (14)$$

2.2.3 Explication des contraintes :

La fonction objective (9) vise à minimiser le coût total de l'affectation. Les contraintes (10) garantissent que la demande des clients finaux est satisfaite. Les contraintes (11) sont les contraintes de capacité. (12) garantissent l'équilibre du flux du produit à travers les installations intermédiaires. Enfin, (13) et (14) sont les contraintes d'intégrité.

2.3 Modélisation du problème résultant :

2.3.1 Les paramètre et les variables :

L : Ensemble des niveaux d'affectation,

l : Niveau d'affectation, de 0 à $L - 1$

A_l : Ensemble des arcs (i,j) du niveau l tels que les i sont les fournisseur et les j sont les client du niveau

A_l^+ : Ensemble des j tels qu'ils existent des i de façon à assurer que (i,j) appartient à A_l

A_l^- : Ensemble des i tels qu'ils existent des j de façon à assurer que (i,j) appartient à A_l

K_l : Ensemble des niveaux de capacité au niveau pour les installation(fournisseurs) du niveau l

c_{ij}^l : Coût unitaire d'affectation de l'installation i à j au niveau l

$u_{i,l,k}$: Niveau de capacité k à l'installation i au niveau l

f_i^{lk} : Coût d'ouverture de l'installation i avec une capacité k au niveau l

d_j : Demande du client j

B : Contrainte budgétaire

x_{ij}^l : Quantité de produits transférée de l'installation i au client j au niveau l

y_i^{lk} : Variable binaire indiquant si l'installation i est ouverte au niveau de capacité k au niveau l

2.3.2 le modèle mathématique du problème résultant

$$\text{Min : } \sum_{l=0}^{l-1} \sum_{(i,j) \in A_l^2} c_{ij}^l \cdot x_{ij}^l \quad (15)$$

$$\sum_{i \in A_1^-} x_{ij}^1 = d_j \quad \forall j \in A_1^+ \quad (16)$$

$$\sum_{j \in A_l^+} x_{ij}^l \leq \sum_{k \in K_l} u_{i,l,k} \cdot y_i^{lk} \quad \forall l \in L, \forall i \in A_l^- \quad (17)$$

$$\sum_{k \in K_l} y_i^{lk} \leq 1 \quad \forall l \in L, \forall i \in A_l^- \quad (18)$$

$$\sum_{l \in L} \sum_{i \in A_1^-} \sum_{k \in K_l} y_i^{lk} \cdot f_i^{lk} \cdot Y_i^{lk} \leq B \quad (19)$$

$$\sum_{i \in A_l^-} x_{ij}^l \geq \sum_{p \in A_{l-1}^+} x_{jp}^{l-1} \quad \forall j \in A_l^+, \forall l \in L \setminus \{1\} \quad (20)$$

$$x_{ij}^l \in R^+ \quad \forall l \in L, \forall (i, j) \in A_l^2 \quad (21)$$

$$y_i^{lk} \in \{0, 1\} \quad \forall l \in L, \forall i \in A_l^-, \forall k \in K \quad (22)$$

2.3.3 Explication des contraintes :

- (15) : Minimiser les coûts totaux des affectations effectuées.
- (16) : Garantir que les demandes des clients finaux sont satisfaites.
- (17) : Contrainte de capacité.
- (18) : Limiter l'ouverture du niveau de capacité à un seul niveau.
- (19) : Limite budgétaire.
- (20) : Garantir l'équilibre du flux du produit à travers les installations intermédiaires.
- (21) et (22) : Contraintes d'intégrité.

Chapitre 3

RESOLUTION

Le chapitre de résolution s'érige comme la pierre angulaire de notre étude, mettant en scène la résolution pratique des modèles mathématiques élaborés pour surmonter les obstacles logistiques. À travers l'utilisation synergique de Java avec CPLEX et Python, nous dévoilons la capacité de ces outils à transformer des formulations abstraites en solutions tangibles. Ce volet opérationnel révèle comment les langages de programmation et l'optimisation linéaire convergent harmonieusement pour relever les défis complexes de la logistique.

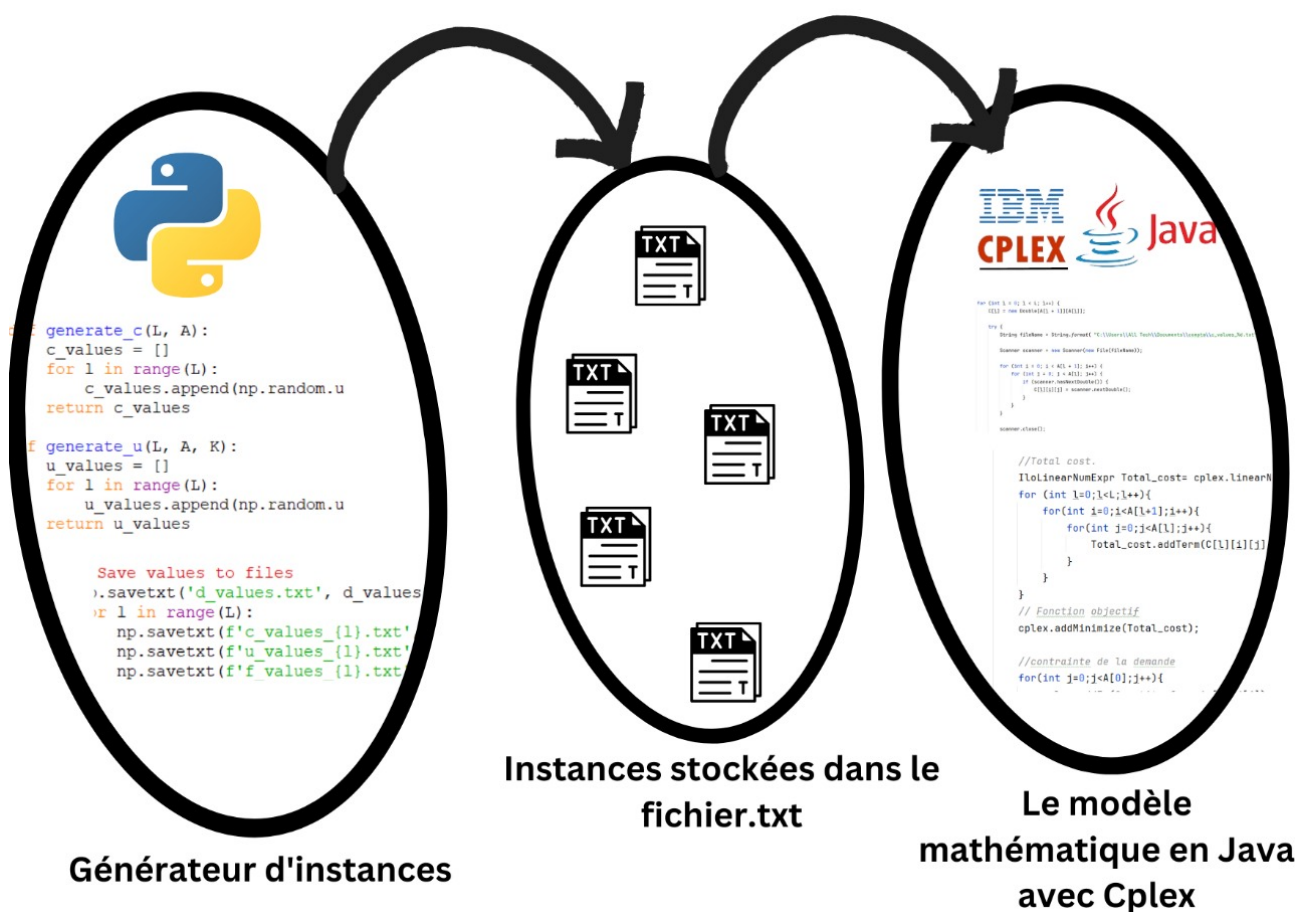


FIGURE 3.1 – Schema illustratif du résolution

La méthodologie de résolution commence par la modélisation mathématique du problème d'optimisation linéaire, intégrant les contraintes spécifiques liées aux niveaux d'affectation et

aux capacités des installations. Un générateur d'instances en Python est ensuite employé, prenant comme entrées les différentes cardinalités des ensembles K_l et A_l . À partir de ces paramètres, le générateur crée des instances du problème, comprenant la demande (d_i), les coûts d'affectation (c_{ij}^l), les niveaux de capacités ($U_i^{l,k}$), et les coûts d'ouverture ($f_i^{l,k}$). Ces instances sont ensuite écrites dans des fichiers texte distincts. Le modèle est implémenté en Java avec CPLEX, et à chaque exécution, il prend en compte ces instances à partir des fichiers texte correspondants pour résoudre automatiquement le problème. Les résultats obtenus sont ensuite analysés, offrant une validation du modèle à travers différentes instances générées par le générateur, et assurant la robustesse et la précision de la solution dans des scénarios variés.

3.1 Techniques et outils utilisés

3.1.1 Java et CPLEX



FIGURE 3.2 – IBM-CPLEX-Logo

CPLEX, développé par IBM, est un puissant solveur d'optimisation linéaire et mixte en nombres entiers, utilisé pour résoudre des problèmes complexes dans divers secteurs. Il se distingue par sa performance élevée sur des problèmes de grande taille et sa capacité à traiter des contraintes multiples. CPLEX est largement utilisé pour sa précision dans la recherche de solutions optimales dans des domaines tels que la logistique, la finance, et la planification. Son intégration avec divers langages de programmation en fait un outil polyvalent pour la modélisation et la résolution d'optimisation.



FIGURE 3.3 – Java-Logo

Java est un langage de programmation polyvalent, orienté objet, qui est largement utilisé pour le développement d'applications logicielles. Il offre une portabilité élevée, permettant l'exécution du code sur différentes plateformes sans nécessiter de modifications majeures.

CPLEX avec Java offre une interface permettant d'intégrer le puissant solveur d'optimisation CPLEX dans des applications Java. Cette combinaison permet aux développeurs de modéliser et résoudre des problèmes complexes d'optimisation linéaire, mixte en nombres entiers, et quadratique, tout en tirant parti de la performance de CPLEX. L'intégration avec Java offre une flexibilité pour incorporer la solution d'optimisation dans des applications Java, facilitant ainsi le déploiement de solutions efficaces et précises pour des défis d'optimisation variés.

3.1.2 Python

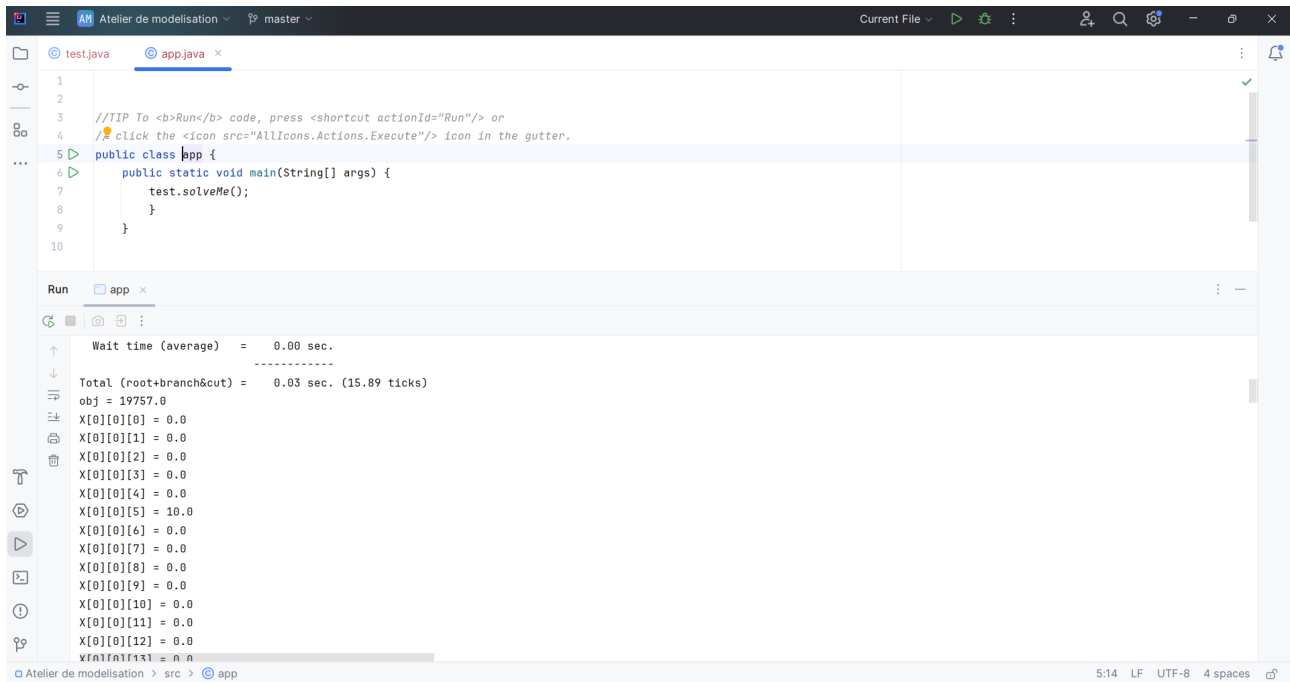


FIGURE 3.4 – Python-Logo

Python est un langage de programmation interprété, polyvalent et haut niveau, reconnu pour sa syntaxe claire et sa facilité d'apprentissage. Dans notre projet, Python est utilisé pour créer un générateur d'instances. Python offre une manipulation de données efficace, facilitant la création d'instances de test diverses pour évaluer la robustesse et l'efficacité du modèle d'optimisation.

3.2 Réalisation :

3.2.1 Résolution par les instances du 1er test :



```

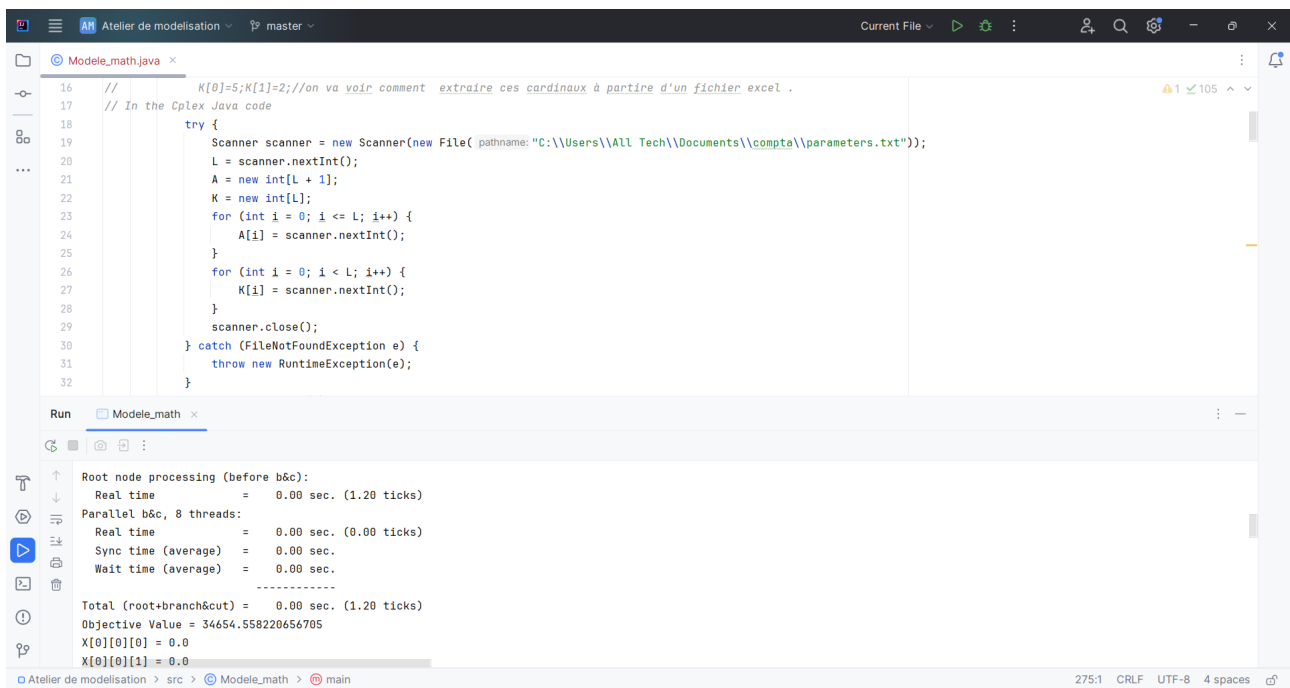
1 //TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
2 // click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
3
4 public class app {
5     public static void main(String[] args) {
6         test.solveMe();
7     }
8 }
9
10
Run app x
Wait time (average) = 0.00 sec.
-----
Total (root+branch&cut) = 0.03 sec. (15.89 ticks)
obj = 19757.0
X[0][0][0] = 0.0
X[0][0][1] = 0.0
X[0][0][2] = 0.0
X[0][0][3] = 0.0
X[0][0][4] = 0.0
X[0][0][5] = 10.0
X[0][0][6] = 0.0
X[0][0][7] = 0.0
X[0][0][8] = 0.0
X[0][0][9] = 0.0
X[0][0][10] = 0.0
X[0][0][11] = 0.0
X[0][0][12] = 0.0
x f a l s e

```

FIGURE 3.5 – Résultats du 1er test avec B=10000.

NB : Les résultats détaillés des tests sont disponibles dans les annexes dédiées à cet effet.

3.2.2 Résolution par les instances du générateur :



```

16 // K[0]=5;K[1]=2;//on va voir comment extraire ces cardinaux à partir d'un fichier excel .
17 // In the Cplex Java code
18 try {
19     Scanner scanner = new Scanner(new File( pathname: "C:\\Users\\All Tech\\Documents\\compte\\parameters.txt"));
20     L = scanner.nextInt();
21     A = new int[L + 1];
22     K = new int[L];
23     for (int i = 0; i <= L; i++) {
24         A[i] = scanner.nextInt();
25     }
26     for (int i = 0; i < L; i++) {
27         K[i] = scanner.nextInt();
28     }
29     scanner.close();
30 } catch (FileNotFoundException e) {
31     throw new RuntimeException(e);
32 }
Run Modele_math x
Root node processing (before b&c):
Real time = 0.00 sec. (1.20 ticks)
Parallel b&c, 8 threads:
Real time = 0.00 sec. (0.00 ticks)
Sync time (average) = 0.00 sec.
Wait time (average) = 0.00 sec.
-----
Total (root+branch&cut) = 0.00 sec. (1.20 ticks)
Objective Value = 34654.558220656705
X[0][0][0] = 0.0
X[0][0][1] = 0.0

```

FIGURE 3.6 – Résultats par les instances du générateur

Dans les deux cas de figure on constate clairement la nature de l'outputs qui est la valeur optimale et les valeurs des variables de decision

Conclusion générale et perspectives :

En résumé, ce projet a souligné l'importance cruciale de la modélisation mathématique, en utilisant CPLEX et Java, pour résoudre des problèmes logistiques complexes via une approche d'optimisation linéaire traitant des défis réels de localisation multi-capacité et multi-niveau.

Malgré les avancées significatives réalisées dans ce projet, certaines perspectives n'ont pas pu être pleinement explorées en raison de contraintes temporelles. L'une de ces perspectives importantes concerne le développement d'une interface utilisateur conviviale et la mise en place d'une simulation du système modélisé.

Initialement prévues dans le cadre de ce projet, ces fonctionnalités devaient permettre une interaction intuitive avec le système d'information élaboré. L'interface utilisateur aurait facilité la visualisation des résultats obtenus, offrant ainsi une expérience utilisateur plus immersive et compréhensible. De plus, la simulation du système aurait permis de tester le modèle dans des scénarios dynamiques.

Bien que ces aspects n'aient pas pu être pleinement réalisés dans le cadre de ce projet, il est important de souligner que cela ne constitue pas une limitation permanente. Les perspectives futures incluent la poursuite du développement de l'interface utilisateur et la mise en œuvre d'une simulation approfondie. Ces ajouts permettront non seulement de rendre le système plus convivial pour les utilisateurs finaux, mais également d'explorer davantage sa performance dans des contextes opérationnels variés. Ainsi, malgré les contraintes actuelles, ces développements demeurent envisageables dans le prolongement de ce projet.

En définitive, ce projet a démontré la pertinence et la puissance de l'optimisation linéaire dans la résolution de problèmes logistiques complexes. Les connaissances acquises et les résultats obtenus fournissent une base solide pour des développements ultérieurs, visant à rendre le système encore plus performant et adaptable aux défis logistiques du monde réel.

Bibliographie

- [1] [<https://stackoverflow.com/>](https://stackoverflow.com/).
- [2] Ibm ilog cplex optimization studio 12.9.0 documentation. <https://www.ibm.com/docs/en/icos/12.9.0?topic=c-ilocplex-4>.
- [3] JetBrains support. <https://www.jetbrains.com/support/>.
- [4] Python developer's guide. <https://devguide.python.org/>.