



Mémoire de fin d'étude

Pour l'obtention du diplôme de Master

Filière : Génie industriel  
Spécialité : Management industriel et logistique

Présenté par : ILLOUL Youcef

Thème

**Réalisation d'une application pour le  
contrôle et l'optimisation des coûts du  
département Logistique Collecte de  
l'entreprise DDA**

Soutenu publiquement, le 13 / 09 / 2020, devant le jury composé de :

M. SOUIR Mehdi	M.C.A	ESSA.Tlemcen	Président
M. BRAHAMI Mustapha Anwar	M.A.A	ESSA.Tlemcen	Directeur de mémoire
M. M. MALIKI Fouad	M.C.B	ESSA.Tlemcen	Co- Directeur de mémoire
M. BENNEKROUF Mohammed	M.C.B	ESSA.Tlemcen	Examineur 1
M. MEKAMCHA Khalid	M.C.B	UAB. Tlemcen	Examineur 2

*Je dédie ce modeste travail :*

*A mes chers parents*

*A qui je dois ce que je suis*

*A ceux qui m'ont tout donné sans rien en retour*

*A toute ma famille*

*Que dieu leur préserve longue vie*

*A tous mes amis ainsi qu'à tous ceux qui me sont chers*

*Que ce travail soit le témoignage sincère et affectueux*

*De ma profonde reconnaissance pour tout ce que vous avez Fait pour moi.*

*Youcef*

## **REMERCIEMENT :**

*Avant toute personne, je remercie le bon Dieu de nous avoir prêté vie, santé et volonté pour achever ce modeste travail.*

*Je tiens à remercier mes encadrants M. BRAHAMI Mustapha Anwar et M. MALIKI Fouad pour tout le temps qu'ils m'ont consacré, pour leur conseils précieux, pour toute leur aide et leur appui durant la réalisation de ce travail.*

*Je tiens à remercier, mes chers parents pour leur encouragement et soutien.*

*J'adresse également un grand merci à M. Sahnoun M'hammed Maître de Conférences à LINEACT de Rouen, pour son aide et ses conseils précieux tout au long de ce projet.*

*Je tiens à remercier chacun des membres du jury pour nous avoir fait l'honneur d'examiner et d'évaluer notre travail*

L'objectif de ce travail est de développer une application d'aide à la décision pour l'entreprise Danone Djurdjura Algérie (DDA), permettant d'améliorer et d'optimiser les tournées de véhicules des opérations de distribution ou d'approvisionnements. L'application développée et baptisée « VRP-MH » (Vehicle Routing Problem – MetaHeuristics) utilise quatre approches basées sur les métaheuristiques pour la résolution des problèmes de tournées complexes intégrant des contraintes temporelles. La conception et la spécification de l'application est accomplie à l'aide du langage de modélisation UML, tandis que l'implémentation est réalisée en utilisant le langage Java sous l'environnement Eclipse.

Mots clés : Application VRP-MH, Outil d'aide à la décision, Tournée de véhicule, Métaheuristiques, UML, Java

This work aims to develop a decision support application for the company Danone Djurdjura Algeria (DDA), that can improve and optimize the routes of vehicles for distribution or supply operations. The developed application is called "VRP-MH" (Vehicle Routing Problem - MetaHeuristics) uses four approaches based on metaheuristics for the resolution of complex routing problems integrating time constraints. The conception and specification of the application is made using UML modeling language, while the implementation is done using the Java language in the Eclipse environment.

Keywords: VRP-MH application, decision support tool, vehicle routing, metaheuristics, UML, Java

الهدف من هذا العمل هو تطوير تطبيق دعم القرارات لشركة Danone Djurdjura Algérie (DDA) ، والذي يمكنه تحسين مسارات المركبات لعمليات التوزيع و الإمداد. يُطلق على التطبيق اسم "VRP-MH" ( Vehicle Routing Problem – MetaHeuristics ) يعتمد على أربعة مناهج تستند على الأدلة العليا لحل مشاكل التوجيه المعقدة التي تدمج القيود الزمنية. يتم تنفيذ مفهوم التطبيق ومواصفاته باستخدام لغة نمذجة UML ، بينما يتم التنفيذ باستخدام لغة Java في بيئة Eclipse .

الكلمات الرئيسية : تطبيق VRP-MH ، أداة دعم القرارات ، توجيه المركبات ، الأدلة العليا (metaheuristics) ، UML ، Java .

## TABLE DES MATIERE

<b>INTRODUCTION GENERALE .....</b>	<b>8</b>
<b>CHAPITRE I.....</b>	<b>10</b>
<b>1. Introduction .....</b>	<b>11</b>
<b>2. La logistique .....</b>	<b>11</b>
<b>3. La chaîne logistique .....</b>	<b>12</b>
3.1. Gestion de la chaîne logistique .....	15
3.2. Structure générale d'une chaîne logistique .....	15
3.3. Niveaux de décisions dans une chaîne logistique .....	16
<b>4. Le transport et la distribution dans la chaîne logistique .....</b>	<b>17</b>
4.1. Le problème du voyageur de commerce TSP .....	18
4.2. Le problème de tournées de véhicules .....	18
4.3. Modélisation mathématique .....	19
4.3.1. Paramètres .....	19
4.3.2. Variables de décision.....	19
4.3.3. Fonction objectif :.....	19
4.3.4. Contraintes.....	20
4.4. Quelques variantes du problème de tournées de véhicules .....	21
<b>5. Les méthodes de résolution du problème VRP : .....</b>	<b>24</b>
5.1. Les méthodes exactes.....	25
5.2. Les méthodes approchées.....	25
5.2.1. Les heuristiques .....	25
5.2.2. Les métaheuristiques .....	26
5.2.3. Les métaheuristiques à solution unique .....	29
5.2.4. Les métaheuristiques à population de solutions .....	32
<b>6. Conclusion.....</b>	<b>34</b>
<b>CHAPITRE II .....</b>	<b>35</b>
<b>1. Introduction .....</b>	<b>36</b>
<b>2. Problématique.....</b>	<b>36</b>

<b>3. Objectifs de l'application VRP-MH.....</b>	<b>37</b>
<b>4. Méthode de conception.....</b>	<b>37</b>
4.1. Définition d'UML (Unified Modeling Language) .....	37
<b>5. Analyse des besoins.....</b>	<b>39</b>
5.1. Identification des acteurs .....	39
5.1.1. Définition d'un acteur : .....	39
5.2. Description textuelle des cas d'utilisations .....	39
5.2.1. Description textuelle du cas d'utilisation « Initialisation des données » .....	40
5.2.2. Description textuelle du cas d'utilisation « Optimisation du problème de tournée » .....	41
<b>6. Conception et implémentation de l'application VRP-MH .....</b>	<b>41</b>
6.1. Diagramme de cas d'utilisation.....	41
6.1.1. Le diagramme de cas d'utilisation associé au Logisticien/Responsable collecte .....	42
6.2. Diagramme de classes.....	42
6.3. Description de l'environnement de développement .....	45
6.3.1. Langage JAVA .....	45
6.3.2. L'IDE Eclipse.....	46
<b>7. Démonstration de l'application VRP-MH .....</b>	<b>46</b>
7.1. L'interface des résultats .....	46
7.2. Exemple d'affichage des résultats par l'application VRP-MH .....	47
7.3. Performances de l'application VRP-MH .....	48
7.3.1. Jeux de donnée .....	48
7.3.2. Réglage des paramètres des métaheuristiques utilisées .....	49
7.3.3. Résultats obtenus .....	50
7.3.4. La validation de résultats de VRP-MH par la méthode exacte Branch & Bound :.....	53
 <b>CONCLUSION GENERALE ET PERSPECTIVES .....</b>	 <b>54</b>
 <b>REFERENCES.....</b>	 <b>55</b>

## TABLE DES FIGURES

Figure 1.1 : Chaîne logistique d'une entreprise de production laitière .....	14
Figure 1.2 : Structure générale d'une chaîne logistique .....	16
Figure 1.3 : Classification de méthodes d'optimisation .....	24
Figure 1.4 : Exemples de solutions voisines dans un problème VRP [NGUEVEU. 2009] .....	28
Figure 1.5 : Algorithme du recuit simulé .....	30
Figure 1.6 : Algorithme de recherche tabou .....	32
Figure 1.7 : Exemples de croisement .....	34
Figure 2.1 : Diagramme de cas d'utilisation associé au Logisticien/Responsable collecte .....	42
Figure 2.2 ; Diagramme de classes de l'application VRP-MH .....	43
Figure 2.3 : L'interface de résultats montrant la solution gloutonne après exécution du programme .....	46
Figure 2.4 : L'interface de résultats montrant la solution de l'algorithme AG+TS .....	47
Figure 2.5 : Figure : Exemple d'une solution de l'algorithme RS .....	47
Figure 2.6 : Exemple d'une solution de l'algorithme AG+TS .....	48

## TABLE DES TABLEAUX

Tableau 2.1 : Description textuelle de cas d'utilisation « Initialisation des données » .....	40
Tableau 2.2 : Description textuelle de cas d'utilisation « Optimisation du problème de tournée » .....	41
Tableau 2.4: Paramètres de l'AG adopté .....	50
Tableau 2.5 : Résultats obtenus par l'application VRP-MH .....	52

## **Introduction générale**

De nos jours, dans un environnement économique de plus en plus concurrentiel, l'optimisation des chaînes logistiques occupe une place importante dans la vie industrielle des entreprises. Le but étant l'amélioration de leur compétitivité en optimisant leurs performances et en réduisant leurs coûts. En effet, la chaîne logistique englobe toutes les tâches réalisées pour la production d'un service ou d'un produit, à savoir l'approvisionnement, la transformation, le stockage, et la distribution. Bien que, la chaîne logistique englobe toutes ces tâches, des études ont montré que les coûts de transport, de l'approvisionnement et de distribution constituent le tiers des coûts opérationnels d'une chaîne logistique. Par conséquent, depuis plusieurs années, un nombre croissant d'entreprises et de chercheurs constatent les bénéfices de l'optimisation de la logistique de distribution/approvisionnement et de transport, et elle est devenu un challenge majeur pour eux. Dans ce sens, l'élaboration des tournées de véhicules pour la l'approvisionnement en matières premières et la livraison les produits finis aux clients constitue l'une des activités principales de la logistique de distribution/approvisionnement et de transport. L'optimisation des activités de distribution/approvisionnement consiste à résoudre en partie les problèmes de tournées de véhicules.

Le travail que nous allons présenter dans ce mémoire de Master concerne le problème d'élaboration des tournées de véhicules connu sous l'appellation VRP (Vehicle Routing Problem). Ce problème classique de l'élaboration des tournées consiste à construire des routes avec un coût minimum pour que les véhicules puissent visiter exactement une fois chaque client géographiquement distribué. Le VRP est un sous problème important dans le domaine des systèmes de distribution/approvisionnement et beaucoup d'efforts ont été consacrés en recherche sur divers aspects du VRP.

L'objectif principal à travers ce projet de fin d'étude consiste à développer une application informatique permettant l'optimisation des problèmes de tournées de véhicules intégrant les contraintes temporelles de fenêtres de temps des clients/points de collecte et de durée maximale des



tournées. L'application développée baptisée VRP-MH (Vehicle Routing Problem – MetaHeuristics) utilise quatre approches basées sur les métaheuristiques pour la résolution des problèmes de tournée. Les métaheuristiques implémentées au sein de l'application VRP-MH sont les suivantes : Recuit simulé, Recherche tabou, Algorithme génétique et une dernière approche hybridant un algorithme génétique avec la recherche tabou.

Le présent rapport s'articulera autour de deux chapitres principaux. Le premier chapitre sera consacré à la présentation du problème VRP et ses principales variantes ainsi qu'une synthèse des méthodes de résolution des problèmes de type VRP. Une attention particulière sera accordée aux méthodes de résolution basées sur les métaheuristiques. Dans le deuxième chapitre, nous présentons les étapes de conception de l'application VRP-MH. La modélisation et la conception de l'application sera réalisée à l'aide du langage UML. Nous exposons ensuite les différents outils technologiques utilisés pour l'implémentation de l'application VRP-MH. Nous concluons ce chapitre par une démonstration de l'exécution de l'application VRP-MH en présentant et interprétant les résultats obtenus.

Enfin, nous terminons ce mémoire par une conclusion, où nous évoquerons les principaux apports de ce travail.

## **Chapitre I**

# **Généralités sur les problèmes de tournées de véhicules et leurs résolutions**

## 1. Introduction

Dans ce chapitre, nous rappelons l'ensemble des définitions et des concepts de base liés à la logistique et à la chaîne logistique. Nous nous intéressons particulièrement à une fonction importante de la gestion de la chaîne logistique qui est l'élaboration des tournées de véhicules et qui constitue le thème central de ce projet de fin d'étude. Après avoir présenté le problème de tournée de véhicules VRP, son modèle mathématique de base ainsi que ses principales variantes, nous présentons une synthèse des méthodes de résolution des problèmes de type VRP. Une attention particulière sera accordée aux méthodes de résolution basées sur les métaheuristiques. Ces dernières seront utilisées pour résoudre efficacement un cas pratique d'élaboration de tournée de véhicules qui concerne la collecte de lait frais de l'entreprise DDA.

## 2. La logistique

La logistique est l'activité qui a pour objet de gérer les flux physiques, et les données (informatives, douanières et financières) s'y rapportant, dans le but de mettre à disposition les ressources correspondant à des besoins (plus ou moins) déterminés en respectant les conditions économiques et légales prévues, le degré de qualité de service attendu, les conditions de sécurité et de sûreté réputées satisfaisantes.

Le mot "logistique" a pour origine un mot grec qui indique l'art du raisonnement et du calcul, la notion de logistique a été utilisée depuis l'époque de la guerre mondiale, et on constate, après toutes les définitions attribuées par les spécialistes à ce mot, que la logistique signifie un ensemble de maillons reliés entre eux et qui collaborent pour accomplir un travail bien déterminé. [P5 Thèse Doctorat Oualid GUEMRI]

Pour [L'Association for Supply Chain Management (ASCM)] la logistique est définie comme :

- 1) Dans un contexte industriel, l'art et la science d'obtenir, produire et distribuer les composants et produits au bon endroit et dans les quantités requises.
- 2) Dans un contexte militaire (qui est l'usage le plus fréquent), cela peut aussi inclure les mouvements de personnel.

Dans la définition de la norme AFNOR (norme X 50-600), la logistique est "une fonction dont la finalité est la satisfaction des besoins exprimés ou latents, aux meilleures conditions économiques pour l'entreprise et pour un niveau de service déterminé. Les besoins sont de nature interne (approvisionnement de biens et de services pour assurer le fonctionnement de l'entreprise) ou externe (satisfaction des clients). La logistique fait appel à plusieurs métiers et savoir-faire qui concourent à la gestion et à la maîtrise des flux physiques et d'informations ainsi que des moyens".

### **3. La chaîne logistique**

De nombreux processus de l'entreprise impliquent donc des facettes logistiques, en particulier la Chaîne Logistique (CL) qui va des fournisseurs aux clients. Davantage encore que d'autres activités, la Logistique est caractérisée par un maillage complexe d'actions entre des acteurs faisant souvent des métiers différents, qu'il faut planifier, piloter ou coordonner dans l'espace et dans le temps. Considérée au départ surtout en termes de stocks, la logistique raisonne de plus en plus en termes de flux : flux de planification et ordonnancement des tâches, ordonnancement et circulation (workflow) des activités, objets et documents nécessaires à la chaîne logistique.

[Lee et al., 1993] a défini la chaîne logistique comme étant un " réseau d'installations qui assure les fonctions d'approvisionnement en matières premières, de transformation de ces matières premières en composants puis en produits finis, et de distribution du produit fini vers le client."

Pour [Tsay et al., 1999] la CL se définit comme "un ensemble de deux ou plusieurs entreprises liées par des flux de marchandises, d'informations et financiers."

[Rota- Franz et al., 2001] reconnaît que "La chaîne logistique d'un produit fini se définit comme l'ensemble des entreprises qui interviennent dans les processus d'approvisionnement en composants, de fabrication, de distribution et de vente du produit, du premier des fournisseurs au client ultime".

Bien que ces définitions paraissent similaires, elles mettent en exergue les points essentiels :

- une chaîne logistique se définit par rapport à un produit fini ou à un service offert,
- une chaîne logistique fait intervenir au moins trois organisations,
- les organisations industrielles sont liées par au moins 3 types de flux : des flux de produits, des flux d'informations et des flux financiers,
- les organisations assurent les fonctions d'approvisionnement, de transformation, de distribution et de vente,
- une organisation peut être impliquée dans plusieurs chaînes logistiques,
- une chaîne logistique est dynamique.

[\[http://theses.univ-lyon2.fr/documents/getpart.php?id=lyon2.2006.buzon\\_1&part=109136\]](http://theses.univ-lyon2.fr/documents/getpart.php?id=lyon2.2006.buzon_1&part=109136)

A partir de ces définitions nous pouvons constater que la CL est un réseau de distribution qui assure les fonctions d'acquisition de matériaux, la transformation de ces matériaux en produits intermédiaires et finis, et la distribution de ces produits finis aux clients.

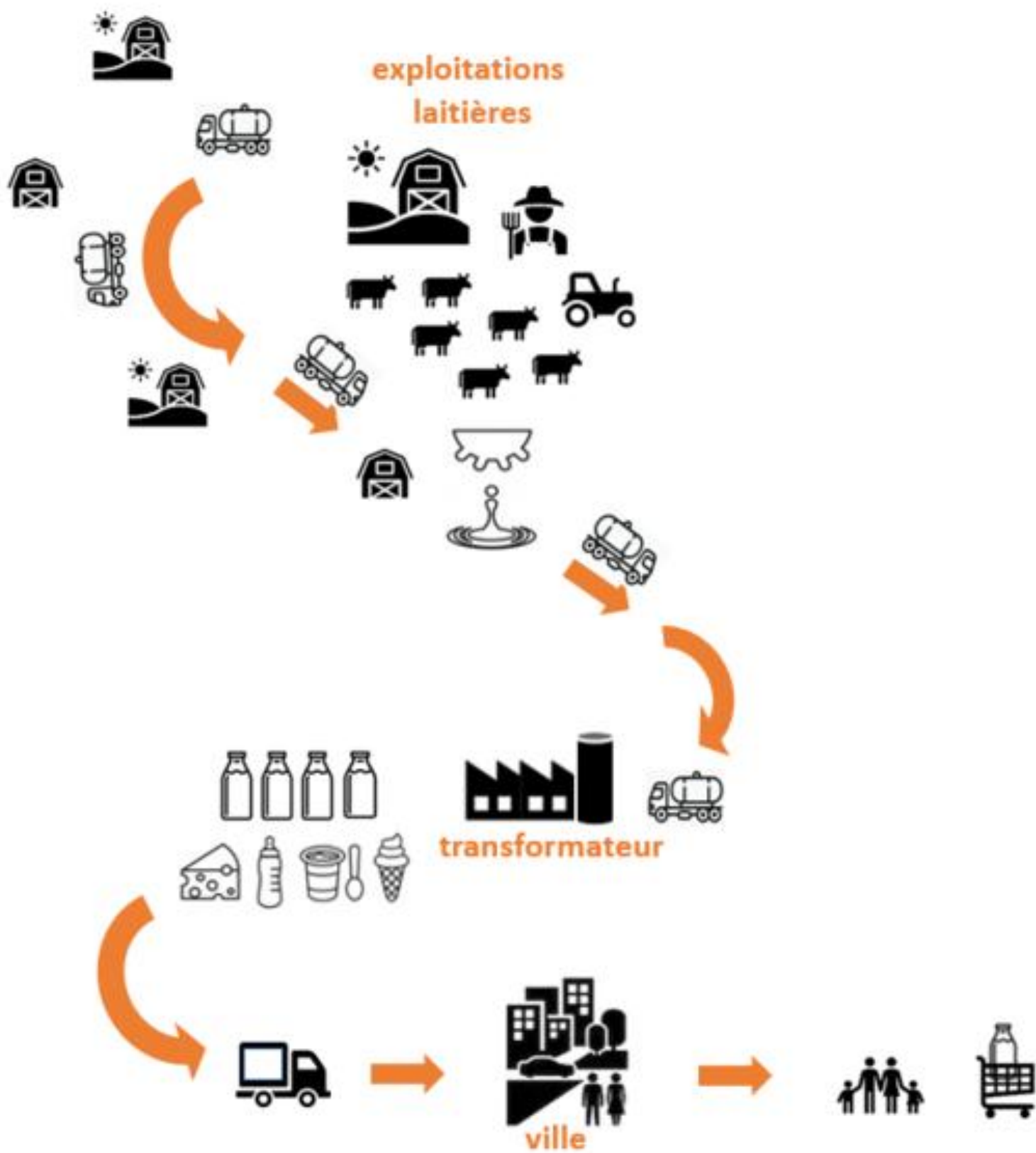


Figure 1.1 : Chaîne logistique d'une entreprise de production laitière

### **3.1. Gestion de la chaîne logistique**

Étant donné la complexité de la chaîne logistique, la gestion de cette dernière (Supply Chain Management (SCM) en anglais) est une tâche qui peut s'avérer difficile et complexe. Le [Council of Supply Chain Management Professionals - CSCMP (2010, p.180)] explique que « *le SCM englobe la planification et la gestion de toutes les activités impliquées dans le sourcing et l'approvisionnement, la conversion, et toutes les activités de gestion logistique. Surtout, elle comprend également la coordination et la collaboration avec les partenaires du canal, qui peuvent être les fournisseurs, les intermédiaires, les tiers fournisseurs de services et les clients. Principalement, le SCM intègre la gestion de l'offre et de la demande au sein et entre les entreprises. Le SCM est une fonction d'intégration avec comme responsabilité principale relier des fonctions et des processus d'affaires importants au sein et entre les entreprises dans un modèle d'affaires cohérent et hautement performant. Il comprend toutes les activités de gestion logistique notées ci-dessus, ainsi que les opérations de fabrication, et il entraîne la coordination des processus et des activités avec et à travers le marketing, les ventes, la conception des produits, les finances et les technologies de l'information* ». Aussi Lambert (2006) explique que « *le Supply Chain management réussi nécessite l'intégration inter fonctionnelle des processus clé au sein de l'entreprise et à travers le réseau des entreprises qui constituent la Supply Chain* ».

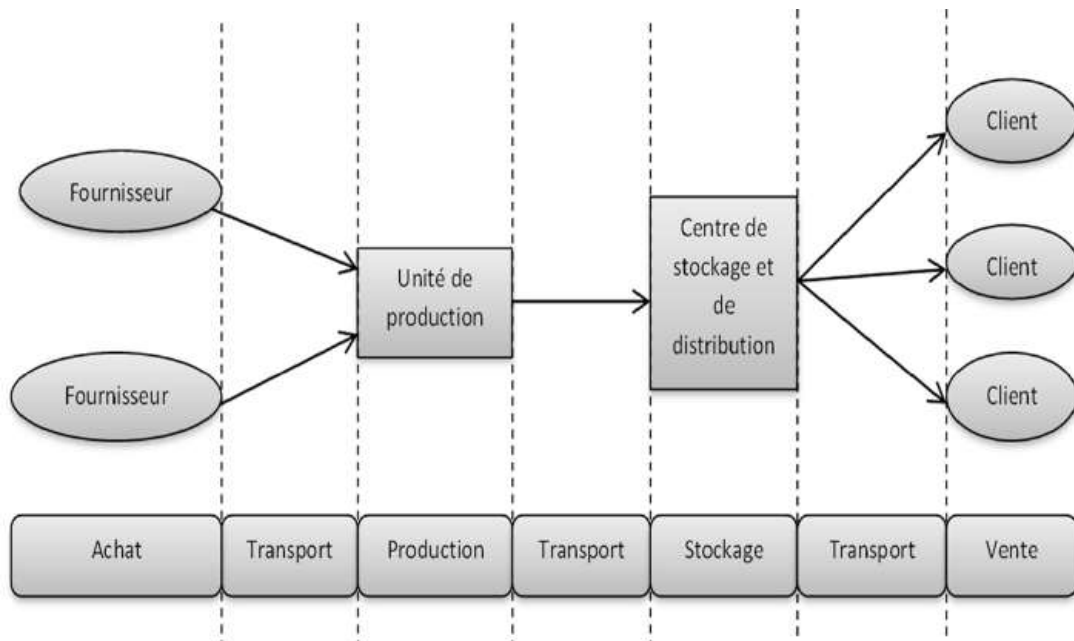
En outre, Mentzer (2004, p.22) définit le SCM comme étant « *la coordination systémique et stratégique des fonctions traditionnelles de gestion au sein d'une entreprise en particulier et à travers les entreprises au sein de la SC, afin d'améliorer la performance à long terme des entreprises individuelles et la SC dans son ensemble.*»

Ainsi, le Supply Chain management est la gestion globale de toutes les activités et processus impliqués dans l'écoulement du produit ou service allant de l'amont jusqu'à l'aval de la chaîne logistique. Visant la maximisation de la valeur globale de cette dernière, il implique principalement la coordination et l'alignement des objectifs et des décisions inter et intra organisationnelles.

### **3.2. Structure générale d'une chaîne logistique**

En prenant la définition de [R. Ganeshan and T. P Harrison 1995], la chaîne logistique a un ensemble de fonctions principales, qui consistent en l'achat des matières premières, la production, le

stockage, la distribution et la vente des produits finis. Dans la *Figure 1.2* nous présentons un exemple d'une chaîne logistique en indiquant l'emplacement de chaque fonction dans la chaîne [F. Galasso. 2007.].



*Figure 1.2 : Structure générale d'une chaîne logistique*

### 3.3. Niveaux de décisions dans une chaîne logistique

La gestion d'une chaîne logistique nécessite de prendre un ensemble de décisions importantes. Dans les chaînes logistiques, on distingue généralement trois niveaux de décisions selon la portée temporelle de la décision: des décisions opérationnelles, des décisions tactiques et des décisions stratégiques [K. Bahloul 2011. ; O. Guemri, and all 2016. ; P. Vallin 2003.]. Nous donnons, ci-après, un bref aperçu sur chaque niveau avec des exemples :

- **Décisions stratégiques** : les décisions stratégiques définissent la politique de l'entreprise sur le long terme, une durée s'étalant souvent sur plusieurs années (deux ans et plus), ce sont les décisions les plus importantes qui permettent de dessiner la structure générale de la chaîne logistique : la localisation des infrastructures (l'ouverture et la fermeture des usines, l'emplacement des dépôts etc.), le choix des partenaires et des fournisseurs et enfin, le choix de mode de transport et des différentes technologies.



- **Décisions tactiques** : leurs influences portent sur un horizon à moyen terme de plusieurs semaines jusqu'à plusieurs mois. A titre d'exemples pour les décisions de ce niveau : l'allocation des clients aux entrepôts, l'affectation des produits aux sites de production et la planification des processus de production.
- **Décisions opérationnelles** : leurs influences portent sur un horizon d'un jour jusqu'à une semaine : l'élaboration des tournée de véhicules pour livrer les produits aux clients ou s'approvisionner en terme de matières premières, l'affectation des ressources aux tâches au sein des unités de production etc.

Dans ce travail nous nous intéressons à un type particulier de décision opérationnelle qui concerne l'élaboration de la tournée de véhicule pour la collecte de lait frais pour l'entreprise Danone Djurdjura Algérie (DDA).

Dans la section suivante nous montrons l'importance des activités de transport et de distribution dans la chaîne logistique en insistant notamment sur la fonction de l'élaboration des tournées de véhicules qui constitue le thème central de notre travail de PFE.

#### **4. Le transport et la distribution dans la chaîne logistique**

L'optimisation des systèmes de transport et de distribution est un défi majeur pour les entreprises car les études ont montré que les coûts de transport et de distribution constituent le tiers des coûts opérationnels d'une chaîne logistique [Journal of business logistics]. Le système de distribution est composé d'un ensemble de fonctions principales, comprenant la localisation des centres d'entreposage et de distribution, la gestion de stock et enfin l'élaboration des tournées de véhicules pour l'approvisionnement en matière premières et la livraison des produits finis aux clients. Donc, l'optimisation des systèmes de distribution consiste à résoudre l'ensemble des problèmes liés à ces fonctions. Dans ce qui suit, nous allons nous intéresser uniquement à la fonction l'élaboration des tournées de véhicules qui constitue le thème principal de ce projet de fin d'étude.

#### **4.1. Le problème du voyageur de commerce TSP**

Le TSP ou Traveling Salesman Problem, est un problème d'optimisation combinatoire qui consiste à trouver le plus court chemin à parcourir pour un seul véhicule, depuis un point de départ, passant par un ensemble de points où chaque point est visité une et une seule fois et retournant au même point de départ.

Le terme problème du voyageur de commerce, vient de la traduction de l'anglais américain Traveling salesman problem, qui est apparu dans les années 1930 ou 40, sans doute à l'université de Princeton où plusieurs chercheurs s'y intéressaient (The traveling Salesman Problem : A Computational Study, Princeton University Press, 2006.).

#### **4.2. Le problème de tournées de véhicules**

Le problème de tournées de véhicules (appelé VRP en anglais pour Vehicle Routing Problem) est une classe de problèmes de recherche opérationnelle et d'optimisation combinatoire, il a été introduit pour la première fois par Dantzig et al. [1954] sous le nom de « Truck Dispatching Problem ». Le but est de minimiser le coût de livraison des biens. Vu son importance et ses multiples applications dans l'industrie et dans la vie quotidienne, le VRP sollicite l'attention de plusieurs chercheurs et fait l'objet de nombreux travaux de recherche.

Le problème VRP est une extension du problème du voyageur du commerce (TSP) [Dhaenens et al., 2002]. Dans sa version la plus basique dite Capacitated VRP (CVRP) ou VRP avec contraintes de capacité, une flotte de véhicules de capacité finie, basée dans un dépôt, doit assurer des tournées entre plusieurs clients (ou villes) ayant demandé chacun une certaine quantité de marchandises. L'ensemble des clients visités par un véhicule désigne la tournée de celui-ci. Chaque client doit être desservi une et une seule fois et chaque tournée commence et se termine au dépôt.

L'objectif du VRP est de minimiser le coût total, c-à-d la somme des distances ou des temps de parcours des tournées, tout en respectant la contrainte de capacité des véhicules : la quantité de marchandises livrées ou collectées sur une tournée ne doit pas dépasser la capacité du véhicule qui l'assure.

### 4.3. Modélisation mathématique

Nous présentons ici une formulation mathématique pour le problème VRP. Dans cette formulation l'objectif est de minimiser la distance parcourue par les véhicules. Au départ, nous définissons les paramètres et les variables utilisées :

#### 4.3.1. Paramètres

- $I = \{0, \dots, N\}$  : ensemble des points de collecte (clients) (dont le dépôt est le client 0)
- $K = \{1, \dots, V\}$  : ensemble des véhicules
- $h_{ij}, \forall (i, j) \in I \times I$  : coût relatif au déplacement du client  $i$  vers le client  $j$
- $d_i, i \in I$  : demande du client  $i$
- $C_k, k \in K$  : capacité du véhicule  $v$

#### 4.3.2. Variables de décision

$$x_{ijk} = \begin{cases} 1 & \text{si le véhicule } k \text{ va de } i \text{ vers } j \\ 0 & \text{sinon} \end{cases} \quad \forall (i, j) \in I \times I, \forall k \in K$$

$$y_{ik} = \begin{cases} 1 & \text{si le véhicule } k \text{ visite le client } i \\ 0 & \text{sinon} \end{cases} \quad \forall i \in I, \forall k \in K$$

#### 4.3.3. Fonction objectif

En utilisant les notations et les hypothèses ci-dessus, la formulation mathématique du modèle VRP est présentée comme suit:

$$(VRP) \quad \text{Minimiser} \quad \sum_{i \in I} \sum_{j \in I} \left( h_{ij} \times \sum_{k \in K} x_{ijk} \right) \quad (1)$$

#### 4.3.4. Contraintes

$$\sum_{k \in K} y_{ik} = 1, i = 1, \dots, N \quad (2)$$

$$\sum_{j \in I} x_{ijk} = y_{ik}, \forall i \in I, \forall k \in K \quad (3)$$

$$\sum_{i \in I} x_{ijk} = y_{jk}, \forall j \in I, \forall k \in K \quad (4)$$

$$\sum_{i \in I} d_i y_{ik} \leq C_k, \forall k \in K \quad (5)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ijk} \leq \text{Card}(S) - 1, \forall S \subset I, 1 \leq \text{Card}(S) \leq \left\lceil \frac{N}{2} \right\rceil, \forall k \in K \quad (6)$$

$$\begin{aligned} x_{ijk} &\in \{0,1\}, \quad \forall i, j \in I, \forall k \in K \\ y_{ik} &\in \{0,1\}, \quad \forall i \in I, \forall k \in K \end{aligned} \quad (7)$$

Dans le modèle proposé, la fonction objectif (1) à minimiser représente la somme des distances parcourues par la flotte de véhicules.

La contrainte (2) indique que chaque client (hors dépôt) est visité par un et un seul véhicule. Les contraintes (3) et (4) expriment que si un véhicule visite un client, il arrive à partir d'un autre client et repart vers un autre client. La contrainte (5) garantit le non dépassement des capacités des véhicules. La contrainte (6) permet d'éliminer les sous-cycles ou les sous-circuits.

Enfin, la contrainte (11) exprime les contraintes d'intégrités des variables de décisions.

Le VRP peut être formulé comme un problème de programmation linéaire en nombre entier. Il a des applications directes dans le transport, la logistique, etc. Par exemple trouver le chemin le plus court pour les bus de ramassage scolaire, pour les camions de ramassage des ordures et aussi pour les camions de collecte de lait. Le VRP, qui est une extension du TSP, présente plusieurs caractéristiques communes aux problèmes d'optimisation combinatoire et fournit ainsi une plateforme idéale pour étudier les méthodes générales applicables à un grand nombre de ces problèmes.

#### 4.4. Quelques variantes du problème de tournées de véhicules

Durant les cinq dernières décennies, de nombreuses variantes de VRP ont été proposées [G. Laporte 2009.]. En effet, l'apparition d'une nouvelle variante est due essentiellement à l'ajout d'une contrainte, une caractéristique ou un nouvel élément au problème (l'ajout, la modification ou l'élimination d'une ou de plusieurs contraintes du VRP peut nous mener vers d'autres variantes). Dans cette partie nous essayons de présenter les variantes les plus connus du VRP [GUEMRI, 2017] :

1. Problème de tournées de véhicules avec contrainte de capacité (Capacitated Vehicle Routing Problem (CVRP)):

Le CVRP est un VRP où la contrainte de capacité de véhicules est imposée, c'est-à-dire que la somme des demandes des clients servis par chaque véhicule ne doit pas dépasser sa capacité. Dans le CVRP la flotte de véhicules est supposée homogène.

2. Problème de tournées de véhicules dynamique (Dynamic Vehicle Routing Problem (DVRP)) :

Le DVRP est une variante du VRP où les données nécessaires pour résoudre le problème dépendent du temps, c-à-dire, soit elles ne sont pas connues à l'avance (manque d'information avant la résolution de problème) ou bien elles peuvent être changées après le commencement des tournées comme par exemple l'ajout d'une nouvelle demande pour un nouveau client ou l'annulation d'une demande d'un client.

3. Problème de tournées de véhicules avec fenêtres de temps (Vehicle Routing Problem with Time Windows (VRPTW)) :

Le VRPTW est un VRP où une contrainte temporelle est imposée. En effet, dans le VRP un véhicule peut visiter un client à n'importe quel moment. Cependant, dans le VRPTW un véhicule ne peut visiter un client que dans un intervalle de temps bien précis, c'est ce que l'on appelle une fenêtre de temps.

4. Problème de tournées de véhicules multi-périodes (Periodic Vehicle Routing Problem (PVRP)) :

Dans le PVRP, nous avons un ensemble de périodes  $P = \{1, \dots, M\}$  qui représente l'horizon de planification, et pour chaque client, nous avons un nombre de visites  $k$  ( $1 \leq k \leq M$ ) qui doivent être

effectuées. Donc, pour résoudre le PVRP nous devons, pour chaque période, déterminer les clients à visiter et construire les tournées de véhicules pour satisfaire ces clients.

5. Problème de tournées de véhicules multi-dépôts (Multi-Depots Vehicle Routing Problem (MDVRP)) :

Dans le VRP, la tournée de chaque véhicule commence et se termine à partir d'un dépôt central. Cependant, dans le MDVRP nous disposons de plusieurs dépôts et dans chaque dépôt nous disposons d'un ensemble de véhicules. Comme dans le VRP, la tournée de chaque véhicule commence et se termine à partir de son dépôt et chaque client doit être servi exactement une seule fois.

6. Problème de tournées de véhicules ouvert (Open Vehicle Routing Problem (OVRP)) :

Contrairement au VRP où la tournée de chaque véhicule doit se terminer au dépôt central, dans le problème OVRP les véhicules ne sont pas obligés de retourner au dépôt central. En effet, cette caractéristique est importante parce qu'elle change complètement la structure de VRP.

7. Problème de tournées de véhicules avec multi-tournées pour chaque véhicule (Le Vehicle Routing Problem with Multiple Trips (VRPMT)) :

La différence entre le VRP et le VRPMT réside dans le fait que dans le VRPMT un véhicule peut faire plusieurs tournées, c'est-à-dire, dans un horizon de planification avec plusieurs périodes et lorsque nous disposons d'une flotte de véhicules limitée, un véhicule peut faire, dans une même période, plusieurs tournées en tenant compte d'une durée maximale d'utilisation par période.

8. Problème de tournées de véhicules stochastique (Stochastic Vehicle Routing Problem (SVRP)) :

Un VRP est stochastique si un des éléments du problème est aléatoire, comme par exemple : la représentation des demandes des clients, du temps de transport sur un arc ou bien du coût d'un arc par une variable aléatoire.

9. Problème de tournées de véhicules multi-compartiments (Multi-Compartment Vehicle Routing Problem (MCVRP)) :

Le MCVRP est utilisé pour modéliser les cas où les demandes des clients concernent plusieurs produits incompatibles. Ces produits doivent être transportés sur un même véhicule mais dans des compartiments indépendants.

*10. Problème de tournées de véhicules avec une flotte hétérogène (Heterogeneous Fleet Vehicle Routing Problem (HVRP)) :*

Dans le VRP nous disposons d'une flotte de véhicules homogènes. Cependant dans le HVRP, nous disposons d'une flotte de véhicules ayant des caractéristiques différentes. Les véhicules peuvent être différenciés par : leurs capacités, leurs coûts de routage, leurs coûts d'utilisation, etc.

*11. Problème de tournées de véhicules avec sélection (Selective Vehicle Routing Problem (SVRP)) :*

On rencontre le SVRP lorsque nous disposons d'une flotte de véhicules avec une capacité qui n'est pas suffisante pour satisfaire toutes les demandes des clients. Donc, nous ne pouvons servir qu'un sous ensemble de clients sélectionnés à partir de l'ensemble des clients.

*12. Problème de tournées de véhicules avec livraison et ramassage (Vehicle Routing Problem with Pickup and Delivery (VRPPD)) :*

Comme pour le VRP, le problème VRPPD consiste à élaborer un ensemble de tournées pour satisfaire les demandes de clients. Cependant dans le VRPPD, nous distinguons deux types de demande : la livraison d'un produit et le ramassage d'un produit.

*13. Problème de tournées de véhicules avec retours (Vehicle Routing Problem with backhauls (VRPB)) :*

Comme pour le VRPPD, dans cette variante nous distinguons deux types de clients : des livreurs où le véhicule doit faire un ramassage et des receveurs où le véhicule doit faire une livraison. Cependant la différence entre les deux variantes est que dans le VRPB nous visitons tous les receveurs avant de visiter les livreurs.

*14. Problème de tournées de véhicules avec distribution partagée (Split Delivery Vehicle Routing Problem (SDVRP)) :*

On rencontre le SDVRP lorsqu'on a des demandes de clients qui sont supérieures à la capacité du véhicule. Dans ce cas, chaque client peut être servi une ou plusieurs fois selon le besoin, c'est-à-dire, le client peut être affecté à une ou plusieurs tournées.

## 5. Les méthodes de résolution du problème VRP :

Les méthodes de résolution des problèmes d'optimisation NP-difficiles tels que les problèmes de tournées de véhicules sont classés en deux catégories : les méthodes exactes et les méthodes heuristiques. Les méthodes exactes sont conçues pour trouver un optimum du problème. Toutefois, leurs durées de calcul tendent à augmenter exponentiellement avec la taille des instances des problèmes NP-difficiles qu'elles essaient de résoudre. Elles doivent donc être arrêtées prématurément lors de la résolution d'instances de grandes tailles, et ne produisent alors que des minorants appelés bornes inférieures ou des majorants appelés bornes supérieures du coût optimal.

A l'inverse, les méthodes heuristiques sont des algorithmes conçus pour produire une solution réalisable du problème traité, de bonne qualité mais pas forcément optimale, sans nécessiter des temps de calcul importants. Elles comprennent les heuristiques constructives, qui construisent une solution, les recherches locales, qui génèrent une suite de solution de coûts décroissants jusqu'à l'optimum local, et les métaheuristiques, qui contiennent divers mécanismes pour se sortir des optima locaux. Les méthodes exactes et les méthodes heuristiques sont complémentaires, car leurs résultats permettent de s'évaluer mutuellement : un faible écart entre les meilleures bornes inférieures et bornes supérieures signifie que l'optimum est proche, un écart nul prouve que les valeurs sont déjà optimales. Enfin, il est courant d'utiliser les résultats d'une de ces approches pour améliorer la performance de l'autre. [NGUEVEU, 2009].

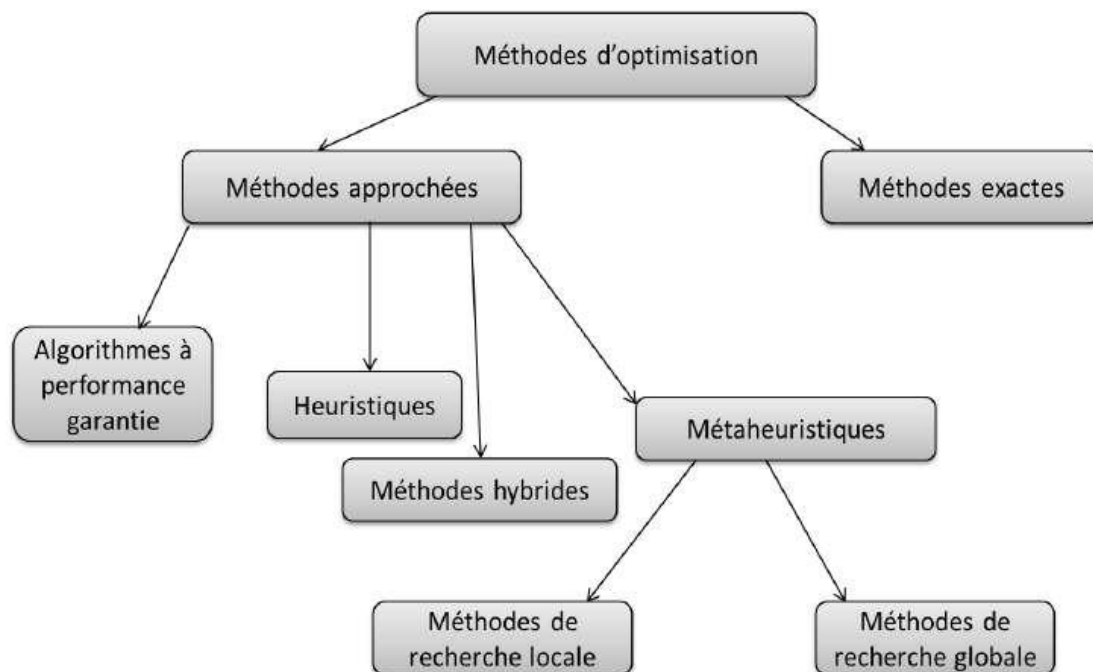


Figure 1.3 : Classification de méthodes d'optimisation



## 5.1. Les méthodes exactes

Les méthodes exactes sont des méthodes qui cherchent la solution optimale d'un problème en explorant exhaustivement toutes les solutions possibles dans l'espace de recherche. Cependant, l'inconvénient majeur de ces méthodes est le temps d'exécution, car toutes les solutions sont évaluées une à une et le temps d'exécution augmente exponentiellement avec la taille du problème considéré. Donc, ces techniques restent inappropriées aux problèmes combinatoires de grandes tailles [GHERBOUDJ. 2013]. Comme exemple de ces méthodes, nous pouvons citer : la programmation dynamique, la génération de colonne, l'algorithme A\*, les algorithmes de type : branch & bound, branch & cut, branch & price, etc. [GUEMRI, 2017]

## 5.2. Les méthodes approchées

Les problèmes d'optimisation du monde industriel sont généralement de grandes tailles et disposent de nombreuses contraintes, donc les méthodes exactes ne sont pas utilisables. Dans ce cas, on doit chercher une bonne solution en un temps raisonnable au lieu d'attendre l'obtention d'une solution optimale après des années de calcul ! [GHERBOUDJ. 2013]. Contrairement aux méthodes exactes, les méthodes approchées (ou méthodes heuristiques) ne garantissent pas l'optimalité de la solution mais, elles permettent de trouver des solutions de très bonne qualité en un temps d'exécution raisonnable, c-à-dire, elles cherchent un bon compromis entre la qualité de la solution et le temps de calcul. Dans la littérature, de nombreuses méthodes approchées ont été proposées. Nous pouvons diviser ces méthodes en 4 catégories: des heuristiques, des métaheuristiques, des algorithmes à performance garantie et des algorithmes hybrides. Par la suite nous présentons chaque catégorie.

### 5.2.1. Les heuristiques

Le mot heuristique vient du grec eurisko εὐρίσκω qui signifie « je trouve » d'où le célèbre Eureka d'Archimède. Dans la littérature, il existe plusieurs définitions d'une heuristique, ici nous présentons celle de Feigenbaum et Feldman (1963) [E. A Feigenbaum and j. Feldman 1963.] : "*Une heuristique (règle heuristique, méthode heuristique) est une règle d'estimation, une stratégie, une astuce, une simplification, ou tout autre type de dispositif qui limite considérablement la recherche de solutions dans des espaces problématiques importants. Les heuristiques ne garantissent pas des solutions*

*optimales. En fait, elles ne garantissent pas une solution du tout. Tout ce qui peut être dit d'une heuristique utile, c'est qu'elle propose des solutions qui sont assez bonnes la plupart du temps".* En outre, dans le domaine de l'optimisation combinatoire on peut dire qu'une heuristique est une méthode approchée développée pour résoudre un problème particulier et elle nécessite des connaissances approfondies sur le problème traité.

Une heuristique, ou méthode approximative, est un algorithme qui fournit rapidement (en temps polynomial) une solution réalisable, pas nécessairement optimale, pour un problème d'optimisation NP-difficile. On oppose les méthodes approchées aux méthodes exactes, qui trouvent toujours l'optimum si on leur en laisse le temps (énumération complète, méthodes arborescentes, programmation dynamique). Les approches heuristiques, contournent le problème de l'explosion combinatoire en faisant délibérément des impasses et n'explorent qu'une partie de l'espace des combinaisons. Une méthode heuristique est généralement conçue pour un problème particulier, en s'appuyant sur sa structure propre. On parle de métaheuristique pour les méthodes approximatives générales, pouvant s'appliquer à différents problèmes.

### **5.2.2. Les métaheuristiques**

Le mot métaheuristique est dérivé de la composition de deux mots grecs : méta, du grec μετά signifiant « au-delà » (ou « à un plus haut niveau ») et heuristique. Les métaheuristiques sont en général des problèmes aux données incomplètes, incertaines, bruitées ou confrontés à une capacité de calcul limitée **mais c'est des algorithmes d'optimisation visant à résoudre des problèmes d'optimisation difficile (souvent issus des domaines de la recherche opérationnelle, de l'ingénierie ou de l'intelligence artificielle) pour lesquels on ne connaît pas de méthode classique plus efficace ou plus robuste.** Ces méthodes sont, pour la plupart, inspirées de la physique (recuit simulé), de la biologie (algorithmes évolutionnaires) ou de l'éthologie (essais particuliers, colonies de fourmis). Le but d'une métaheuristique est de résoudre un problème d'optimisation donné : elle cherche un objet mathématique (une permutation, un vecteur, etc.) minimisant (ou maximisant) une fonction objectif, qui décrit la qualité d'une solution au problème.

L'ensemble des solutions possibles forme l'espace de recherche. L'espace de recherche est au minimum borné, mais peut être également limité par un ensemble de contraintes.

Dans certains cas, le but recherché est explicitement de trouver un ensemble d'optima "satisfaisants". L'algorithme doit alors trouver l'ensemble des solutions de bonne qualité, sans nécessairement se limiter au seul optimum : on parle de méthodes multimodales.

Les métaheuristiques manipulent une ou plusieurs solutions, à la recherche de l'optimum, la meilleure solution au problème. Les itérations successives doivent permettre de passer d'une solution de mauvaise qualité à la solution optimale. L'algorithme s'arrête après avoir atteint un critère d'arrêt, consistant généralement en l'atteinte du temps d'exécution imparti ou en une précision demandée.

D'une manière générale, les métaheuristiques s'articulent autour de ces notions :

- a) Voisinage : Le voisinage d'une solution est un sous-ensemble de solutions qu'il est possible d'atteindre par une série de transformations données. Par extension on désigne parfois par le terme « voisinage » l'ensemble des transformations considérées. Un voisinage simple pour le problème du voyageur de commerce sera, par exemple, l'ensemble des solutions qu'il est possible de construire en permutant deux villes dans une solution donnée (Figure 1.4).
- b) Diversification : La diversification (ou exploration) désigne les processus visant à récolter de l'information sur le problème optimisé.
- c) Intensification : L'intensification (ou exploitation) vise à utiliser l'information déjà récoltée pour définir et parcourir les zones intéressantes de l'espace de recherche.
- d) Apprentissage : La mémoire est le support de l'apprentissage, qui permet à l'algorithme de ne tenir compte que des zones où l'optimum global est susceptible de se trouver, évitant ainsi les optima locaux.

Une solution ou un ensemble de solutions est parfois appelé un état, que la métaheuristique fait évoluer via des transitions ou des mouvements en progressent de façon itérative et alternant des phases d'intensification, de diversification et d'apprentissage, ou en mêlant ces notions de façon plus étroite. Si une nouvelle solution est construite à partir d'une solution existante, elle est sa voisine. Le choix du voisinage et de la structure de donnée le représentant peut être crucial. L'état de départ est souvent choisi aléatoirement, l'algorithme se déroulant ensuite jusqu'à ce qu'un critère d'arrêt soit atteint.

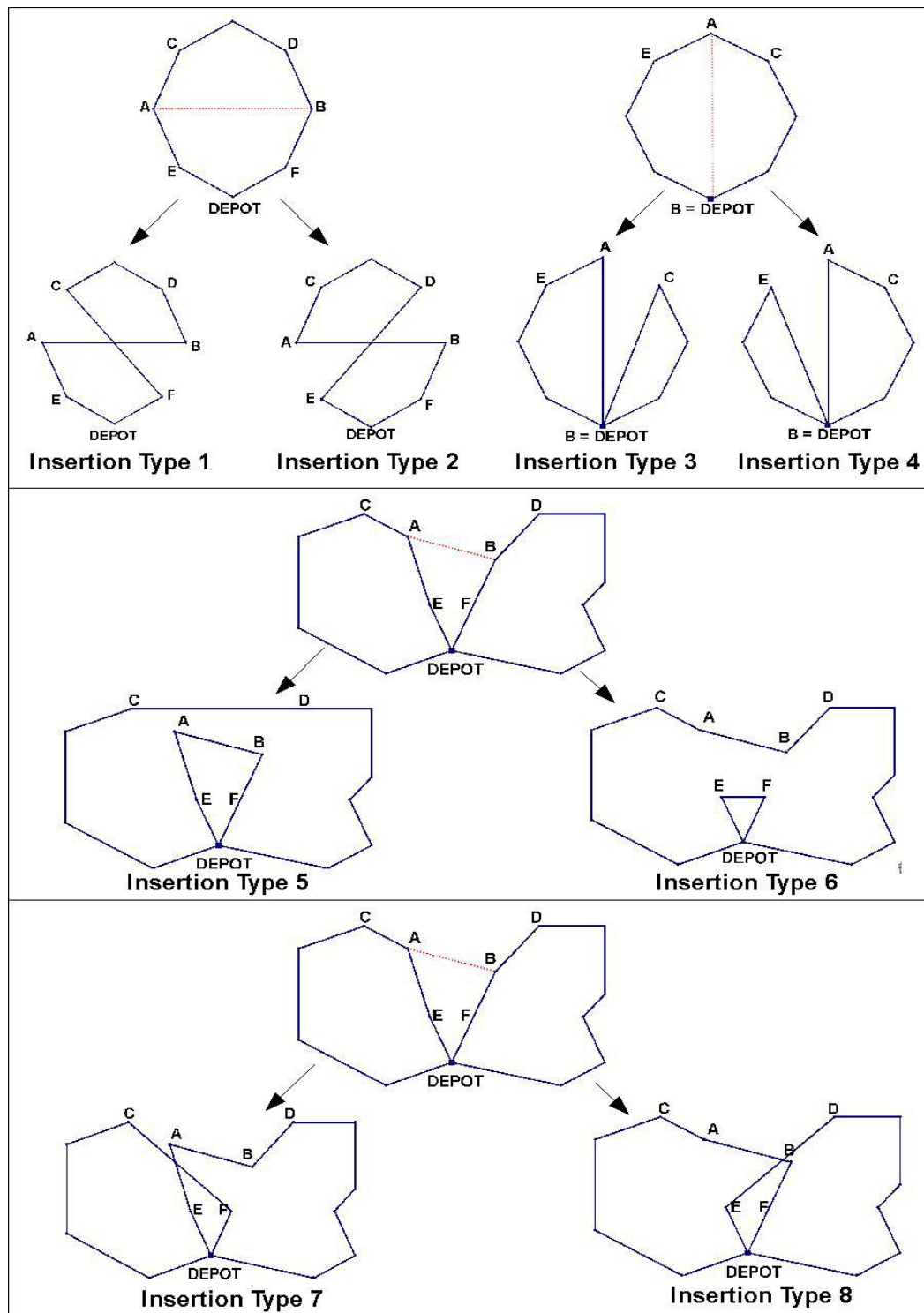


Figure 1.4 : Exemples de solutions voisines dans un problème VRP [NGUEVEU. 2009]

Lorsqu'une solution est associée à une seule valeur, on parle de problème mono-objectif, lorsqu'elle est associée à plusieurs valeurs, on parle de problème multi-objectifs (ou multi-critères).

Dans un numéro spécial de la revue scientifique *European Journal of Operational Research*, consacré aux applications des métaheuristiques, les éditeurs ont constaté que la majorité des 20 articles publiés le furent dans deux domaines : les problèmes d'ordonnancement et de logistique. Les méthodes les plus utilisées appartiennent à la famille des algorithmes évolutionnaires, souvent hybridés avec des méthodes de recherche locale.

Quelques exemples de problèmes concrets, optimisés via des métaheuristiques :

- Problèmes de tournée de véhicules
- Optimisation de réseaux mobiles UMTS
- Gestion du trafic aérien
- Optimisation des plans de chargement des cœurs de réacteurs nucléaires

Dans le cadre de ce projet de fin d'étude, nous proposons quatre métaheuristiques pour la résolution d'une nouvelle variante du problème VRP baptisé « Vehicle Routing Problem with Time Windows and Duration Constraints (VRPTWDC) ». De plus, ces quatre métaheuristiques proposées seront appliquées sur un cas pratique issu de l'industrie agro-alimentaire qui concerne de la collecte de lait frais pour l'entreprise DDA. Ceci fera l'objet des chapitres 3 et 4.

### **5.2.3. Les métaheuristiques à solution unique**

Les méthodes à solution unique, plus connues sous le nom de méthodes de recherche locale ou encore méthodes de trajectoire, sont basées sur l'évolution d'une seule solution dans l'espace de recherche. Typiquement, les méthodes de recherche locale démarrent d'une solution unique, puis à chaque itération, la solution courante est déplacée dans un voisinage local en espérant améliorer la fonction objectif. Les méthodes à solution unique englobent principalement la méthode de descente, la recherche locale itérée, la recherche à voisinage variable, la méthode du recuit simulé et la recherche tabou. Ces méthodes diffèrent essentiellement dans leur manière d'exploiter le voisinage local, i.e, choix des candidats et critères de déplacement.

Dans ce qui suit, nous présentons uniquement les métaheuristiques à solution unique utilisées pour la résolution de notre problème VRPTWDC, à savoir, le recuit simulé et la recherche tabou.

#### 4.3.2.1. Recuit simulé

Le recuit simulé est une méthode de programmation empirique (métaheuristique) inspirée d'un processus utilisé en métallurgie. On alterne dans cette dernière des cycles de refroidissement lent et de réchauffage (recuit) qui ont pour effet de minimiser l'énergie du matériau. Cette méthode est transposée en optimisation pour trouver les extrema d'une fonction.

Elle a été mise au point par trois chercheurs de la société IBM, S. Kirkpatrick, C.D. Gelatt et M.P. Vecchi en 1983, et indépendamment par V. Černý en 1985.

Le recuit simulé s'appuie sur l'algorithme de Metropolis-Hastings, qui permet de décrire l'évolution d'un système thermodynamique. Par analogie avec le processus physique, la fonction objectif à optimiser est assimilée à l'énergie du matériau  $E$ . On introduit également un paramètre fictif, la température  $T$  du système. La Figure 1.5 détaille l'algorithme du recuit simulé.

**Nécessite :** La fonction objectif  $f$  et la taille de la liste taboue

- 1: Générer une solution aléatoire  $S$
- 2: Calculer la *fitness*  $f(S)$  associée à la solution initiale  $S$
- 3: Initialiser la solution optimale :  $S_{opt} \leftarrow S$
- 4: **Tant que** la condition d'arrêt n'est pas vérifiée **Faire**
- 5:     Générer la liste des candidats non tabous par opération de voisinage
- 6:     Trouver la meilleure solution  $S'$  parmi les candidats
- 7:     **Si**  $f(S') < f(S)$  **Alors**
- 8:          $S \leftarrow S'$
- 9:          $S_{opt} \leftarrow S$
- 10:     **Fin Si**
- 11:     Mettre à jour la liste taboue
- 12: **Fin tant que**
- 13: **Retourner :** La solution optimale  $S_{opt}$

*Figure 1.5 : Algorithme du recuit simulé*

#### **4.3.2.2. Recherche tabou**

La recherche tabou est une métaheuristique d'optimisation présentée par Fred W. Glover en 1986. Cette méthode est une métaheuristique itérative qualifiée de recherche locale au sens large.

L'idée de la recherche tabou consiste, à partir d'une position donnée, à en explorer le voisinage et à choisir la position dans ce voisinage qui minimise la fonction objectif.

Il est essentiel de noter que cette opération peut conduire à augmenter la valeur de la fonction (dans un problème de minimisation) : c'est le cas lorsque tous les points du voisinage ont une valeur plus élevée. C'est à partir de ce mécanisme que l'on sort d'un minimum local.

Le risque cependant est qu'à l'étape suivante, on retombe dans le minimum local auquel on vient d'échapper. C'est pourquoi il faut que l'heuristique ait de la mémoire : le mécanisme consiste à interdire (d'où le nom de tabou) de revenir sur les dernières positions explorées.

Les positions déjà explorées sont conservées dans une file FIFO (appelée souvent liste tabou) d'une taille donnée, qui est un paramètre ajustable de l'heuristique. Cette file doit conserver des positions complètes, ce qui dans certains types de problèmes, peut nécessiter l'archivage d'une grande quantité d'informations. Cette difficulté peut être contournée en ne gardant en mémoire que les mouvements précédents, associés à la valeur de la fonction à minimiser

**Nécessite :** La fonction objectif  $f$ , la température maximale  $T_{max}$ , la température minimale  $T_{min}$  et la fonction de diminution de la température  $abaisser(T)$ .

- 1: Initialiser la température :  $T \leftarrow T_{max}$
- 2: Générer une solution aléatoire  $S$
- 3: Calculer la *fitness*  $f(S)$  associée à la solution initiale  $S$
- 4: Initialiser la solution optimale :  $S_{opt} \leftarrow S$
- 5: **Tant que**  $T > T_{min}$  **Faire**
- 6:     **Tant que** l'équilibre thermodynamique n'est pas atteint **Faire**
- 7:         Tirer une nouvelle solution  $S'$  dans le voisinage de  $S$
- 8:         Calculer la variation d'énergie :  $\Delta f \leftarrow f(S') - f(S)$
- 9:         **Si**  $\Delta f \leq 0$  **Alors**
- 10:             Accepter la nouvelle solution :  $S \leftarrow S'$
- 11:         **Sinon**
- 12:             **Si**  $\exp\left(-\frac{\Delta f}{T}\right) > x$  aléatoire  $\in [0, 1]$  **Alors**
- 13:                 Accepter la nouvelle solution :  $S \leftarrow S'$
- 14:             **Fin Si**
- 15:         **Fin Si**
- 16:         **Si**  $f(S') < f(S)$  **Alors**
- 17:             Mettre à jour la solution optimale :  $S_{opt} \leftarrow S'$
- 18:         **Fin Si**
- 19:     Abaisser la température  $T$  :  $T \leftarrow abaisser(T)$
- 20:     **Fin tant que**
- 21: **Fin tant que**
- 22: **Retourner :** La solution optimale  $S_{opt}$

*Figure 1.6 : Algorithme de recherche tabou*

#### 5.2.4. Les métaheuristiques à population de solutions

Contrairement aux algorithmes partant d'une solution singulière, les métaheuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions.

On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes issus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin [Darwin, 1859] et les algorithmes d'intelligence en essaim qui, de la même manière que les algorithmes évolutionnaires, proviennent d'analogies avec des phénomènes biologiques naturels.

Nous citons quelques algorithmes évolutionnistes :

- les stratégies d'évolution,
- les algorithmes génétiques,
- les algorithmes à évolution différentielle,
- les algorithmes à estimation de distribution,
- les systèmes immunitaires artificiels,
- la recomposition de chemin (Path relinking en anglais)
- Shuffled Complex Evolution (Duan et al. 1992)



Dans la prochaine section, nous présentons les algorithmes génétiques qui seront utilisés comme métaheuristique pour la résolution du problème VRPTWDC étudié dans le cadre de projet de fin d'étude.

#### 4.3.2.3. Les algorithmes génétiques

Les algorithmes génétiques (GA : Genetic Algorithms en anglais) sont, sans conteste, la technique la plus populaire et la plus largement utilisée des algorithmes évolutionnaires. Les origines de ces algorithmes remontent au début des années 1970, avec les travaux de John Holland et ses élèves à l'Université du Michigan sur les systèmes adaptatifs [Holland, 1975]. L'ouvrage de référence de David E. Goldberg [Goldberg, 1989] a fortement participé à leur essor. Ces algorithmes se détachent en grande partie par la représentation des données du génotype, initialement sous forme d'un vecteur binaire et plus généralement sous forme d'une chaîne de caractères.

Chaque étape de l'algorithme génétique (AG) est associée à un opérateur décrivant la façon de manipuler les individus :

- *Sélection* : Pour déterminer quels individus sont plus enclins à se reproduire, une sélection est opérée. Il existe plusieurs techniques de sélection, les principales utilisées sont la sélection par tirage à la roulette (roulette-wheel selection), la sélection par tournoi (tournament selection), la sélection par rang (ranking selection), etc [Goldberg & Deb, 1991; Blicke & Thiele, 1995].

- *Croisement* : L'opérateur de croisement combine les caractéristiques d'un ensemble d'individus parents (généralement deux) préalablement sélectionnés, et génère de nouveaux individus enfants. Là encore, il existe de nombreux opérateurs de croisement, par exemple le croisement en un point, le croisement en n-points ( $n \geq 2$ ) et le croisement uniforme (voir *Figure 1.7*).

- *Mutation* : Les descendants sont mutés, c'est-à-dire que l'on modifie aléatoirement une partie de leur génotype, selon l'opérateur de mutation.

- *Remplacement* : Le remplacement (ou sélection des survivants), comme son nom l'indique, remplace certains des parents par certains des descendants. Le plus simple est de prendre les meilleurs

individus de la population, en fonction de leurs performances respectives, afin de former une nouvelle population (typiquement de la même taille qu'au début de l'itération).

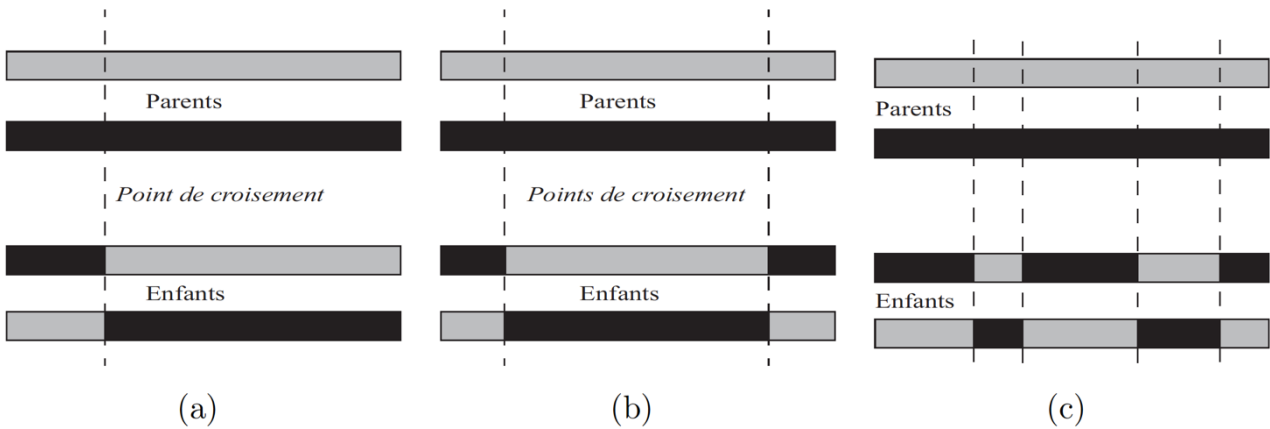


Figure 1.7 : Exemples de croisement : (a) croisement simple en un point, (b) croisement en deux points, (c) croisement uniforme [Boussaid, 2013]

## 6. Conclusion

À travers ce deuxième chapitre nous avons exposé les concepts de la logistique, la chaîne logistique et le Supply Chain Management. Une attention particulière a été accordée à l'élaboration des tournées de véhicules qui constitue l'une des fonctions les plus importantes de la gestion de la chaîne logistique. Nous avons énoncé par la suite la définition du problème de tournée de véhicule VRP et présenté son modèle de base ainsi que ses principales variantes. Nous avons dressé une synthèse des méthodes de résolution des problèmes de type VRP en s'intéressant particulièrement aux métaheuristiques. Enfin, nous avons détaillé le fonctionnement de trois métaheuristiques, à savoir, le recuit simulé, la recherche tabou et les algorithmes génétiques, ces métaheuristiques seront utilisées dans la suite de ce mémoire.

Dans le chapitre qui suit, nous développons un nouveau modèle de tournée de véhicules intégrant deux contraintes temporelles et proposons quatre métaheuristiques pour sa résolution.

## **Chapitre II**

### **Généralités sur les problèmes de tournées de véhicules et leurs résolutions**

## 1. Introduction

Dans ce chapitre, nous présentons les étapes de conception et de développement d'une application d'optimisation de problèmes de tournée de véhicules. L'application développée baptisée VRP-MH (Vehicle Routing Problem – MetaHeuristics) utilise quatre approches basées sur les métaheuristiques pour la résolution des problèmes de tournée. Dans la première partie de ce chapitre, nous présentons la modélisation de l'application en utilisant le langage UML. Nous exposons ensuite les différents outils technologiques utilisés pour l'implémentation de l'application VRP-MH. Enfin, nous terminons ce chapitre par une démonstration de l'exécution de l'application VRP-MH en présentant et interprétant les résultats obtenus.

## 2. Problématique

Le problème VRP avec la contrainte de fenêtres de temps et temps maximal résolu avec l'application VRP-MH est une extension du problème classique de tournée de véhicules VRP. Il modélise bien un problème de transport très répandu qui est celui de la collecte de produits auprès d'un ensemble de points de collecte (clients) répartis géographiquement. Ainsi, le problème VRPTWDC (Vehicle Routing Problem with Time Windows and Duration Constraints) consiste à affecter chaque client à une tournée (ou route) effectuée par un seul véhicule et à trouver un ordre de visites des clients pour chaque véhicule de façon à satisfaire les contraintes de capacité des véhicules, et les quantités de produit livré par chaque client. En plus des contraintes de capacités, le problème VRPTWDC impose deux contraintes temporelles :

- (i) Fenêtres de temps pour chaque client, où un véhicule ne peut visiter un client que dans un intervalle de temps bien précis et
- (ii) Une contrainte sur la durée maximum de chaque route.

L'application VRP-MH nous permettra donc de trouver l'ensemble des tournées qui minimisent la distance totale parcourue pour un nombre minimal de véhicules partant d'un dépôt et y retournant, et aussi d'atteindre les objectifs énumérés dans la sous-section suivante.

### **3. Objectifs de l'application VRP-MH**

Notre travail consiste à développer une application qui a pour but l'optimisation l'amélioration des problèmes VRP en offrant la possibilité d'utiliser quatre 04 approches de résolution. L'application qui vise également à atteindre les objectifs suivants :

- L'application constitue un outil d'aide à la décision qui permet aux décideurs et aux logisticiens d'améliorer leurs tournées de véhicules.
- Optimiser les problèmes de tournées de véhicules de différentes tailles.
- Optimiser les problèmes de type VRP classique et d'autres variantes de ce célèbre problème.
- Améliorer des solutions de tournée de véhicules déjà existantes
- Résoudre les problèmes d'optimisation avec quatre (04) métaheuristiques différentes
- Offrir la possibilité de comparer entre les quatre métaheuristiques
- Illustrer les tournées de véhicules obtenues sous forme graphique.

### **4. Méthode de conception**

Modéliser, c'est décrire de manière visuelle et graphique les besoins et, les solutions fonctionnelles et techniques d'une application.

Pour ce faire, on utilisera le langage UML2 (Unified Modeling Language) qui nous permettra de décrire l'application sous différents angles. [<http://openclassrooms.com/courses/debutez-l-analyse-logicielle-avec-uml>]

#### **4.1. Définition d'UML (Unified Modeling Language)**

UML est un langage visuel constitué d'un ensemble de schémas, appelés des diagrammes, qui donnent à chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter l'application à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par l'application, etc.

Réaliser ces diagrammes revient donc à modéliser les besoins de l'application à développer.

Le langage UML fournit 13 diagrammes, qui sont classés selon deux types de vues:

- Diagrammes structurels ou diagrammes statiques (UML Structure). Parmi ces diagrammes structurels, nous avons choisi d'utiliser le diagramme de classes (Class diagram) dans notre modélisation.
- Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior) dans lesquels nous avons choisi d'utiliser le diagramme de cas d'utilisation (Use case diagram) dans notre modélisation.

Nous avons choisi UML principalement pour trois avantages qui sont [BENACHOUR, TOUATI, 2015]:

1. *Sa souplesse* : Une des forces d'UML est de permettre son utilisation à la carte. En effet, on peut l'utiliser aussi bien pour mettre en œuvre des développements basés sur des méthodes agiles (développement par prototypage, par exemple) que pour des processus de qualité logicielle assez complexes.
2. *Sa cohérence* : La forte utilisation d'UML dans le monde industriel vient de ce que les technologies modernes dérivent majoritairement des approches objet. La raison est double : d'une part, c'est à partir d'elle qu'UML fut construit, et, d'autre part, UML a su évoluer depuis sa création et suivre l'évolution des approches objet vers les approches composant avec l'avènement d'UML 2. UML est donc parfaitement en phase avec les technologies modernes.
3. *Ses performances* : L'utilisation d'UML rend le déroulement d'un projet plus rapide qu'avec des approches de modélisation plus lourdes grâce aux notations graphiques simples et efficaces, les outils qui le supportent ont fait de réels progrès en termes d'aide au développement (notamment à travers l'implémentation de patrons de conception tels que les design patterns), mais aussi grâce à la génération de code ou aux modèles exécutables qui permettent de voir évoluer dynamiquement un modèle.

## **5. Analyse des besoins**

### **5.1. Identification des acteurs**

#### **5.1.1. Définition d'un acteur :**

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié, autrement dit un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages susceptibles d'être porteurs de données. [P.ROQUES, F.VALLÉE, 2003].

Dans le cadre de notre application, nous avons distingué un seul acteur uniquement qui peut représenter le logisticien ou le responsable de l'élaboration des tournées dans une entreprise.

### **5.2. Description textuelle des cas d'utilisations**

Dans ce qui suit, nous allons décrire l'ensemble des cas d'utilisation.

### 5.2.1. Description textuelle du cas d'utilisation « Initialisation des données »

Titre	Initialisation des données
Acteur	Logisticien – Responsable collecte
Description	Initialisation de toutes les données nécessaires à l'optimisation du problème VRP considéré
Précondition	L'application doit être démarrée
Scénario nominal	<ol style="list-style-type: none"> <li>1. Il faut tout d'abord fixer le nombre de nœuds (clients ou points de collecte) et le nombre de véhicules utilisés.</li> <li>2. Fixer pour chaque client/point de collecte : le nom, la localisation, la production /demande et la fenêtre du temps.</li> <li>3. Initialiser la matrice des distances qui représente la distance entre chaque paire de clients/points de collecte.</li> <li>4. Initialiser les données suivantes pour chaque véhicule: <ul style="list-style-type: none"> <li>- L'identifiant</li> <li>- La capacité</li> <li>- La vitesse</li> </ul> </li> <li>5. Fixer la durée maximale pour chaque tourné</li> </ol>

*Tableau 2.1 : Description textuelle de cas d'utilisation « Initialisation des données »*



### 5.2.2. Description textuelle du cas d'utilisation « Optimisation du problème de tournée »

Titre	Optimisation du problème de tournée
Acteur	Logisticien – Responsable collecte
Description	Possibilité d'obtenir la tournée de véhicule en utilisant l'une ou les métaheuristiques proposées par l'application.
Précondition	L'utilisateur doit d'abord initialiser les données.
Scénario nominal	<ol style="list-style-type: none"><li>1. L'utilisateur choisi la ou les métaheuristiques à exécuter.</li><li>2. L'utilisateur peut choisir un jeu de paramètre de la ou les métaheuristique choisies.</li><li>3. Exécuter la ou les métaheuristiques choisies.</li><li>4. L'application affiche les résultats des tournées de véhicules trouvées.</li></ol>

Tableau 2.2 : Description textuelle de cas d'utilisation « Optimisation du problème de tournée »

## 6. Conception et implémentation de l'application VRP-MH

### 6.1. Diagramme de cas d'utilisation

**Définition :** UML propose de représenter les cas d'utilisation sous une forme graphique nommée diagramme de cas d'utilisation. Les rôles sont définis pour chaque acteur. Une relation entre l'acteur et le cas d'utilisation représente une communication. Le cas d'utilisation est représenté par une ellipse qui porte son nom à l'intérieur. Des notes peuvent être portées sur le diagramme afin d'y ajouter des informations.

Puisque nous avons identifié un seul acteur pour notre application, nous aurons un seul diagramme de cas d'utilisation associé à cet acteur (le logisticien ou le responsable de collecte).

### 6.1.1. Le diagramme de cas d'utilisation associé au Logisticien/Responsable collecte

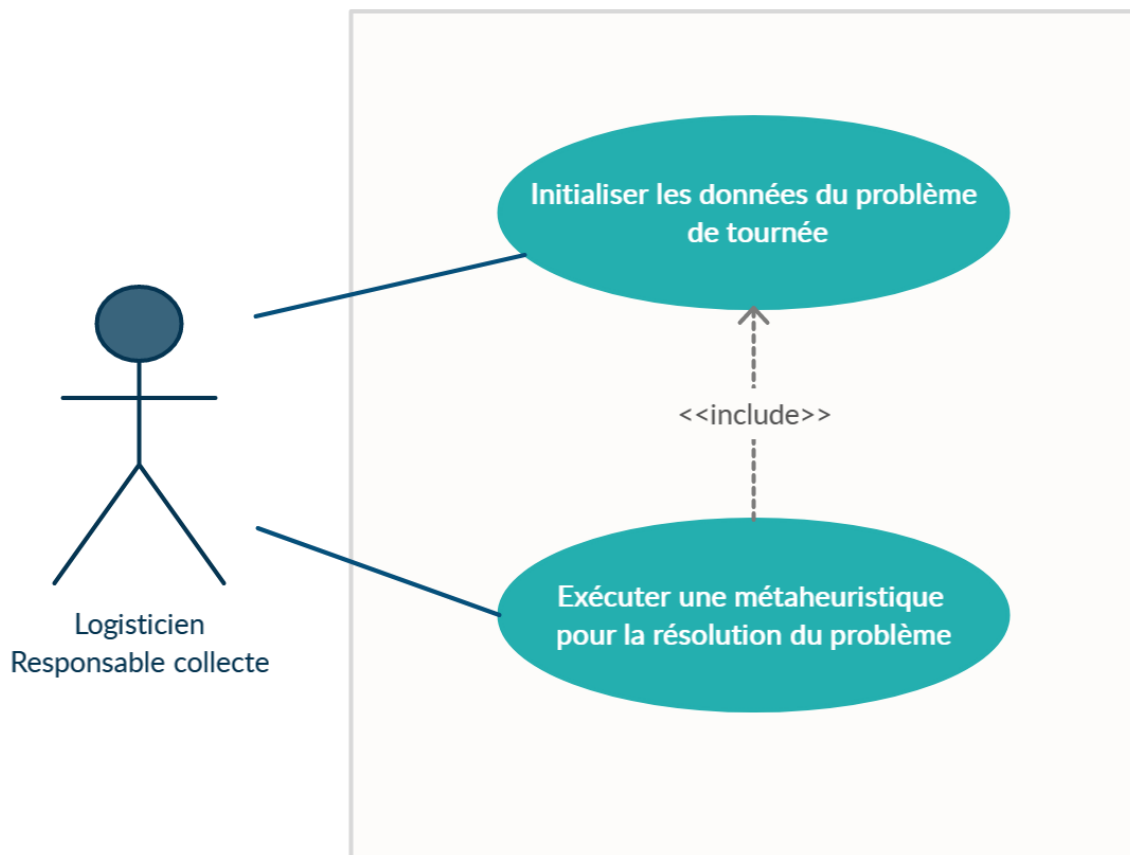


Figure 2.1 : Diagramme de cas d'utilisation associé au Logisticien/Responsable collecte

## 6.2. Diagramme de classes

La caractéristique principale d'une application réside dans sa structure. La structure de l'application est obtenue en partitionnant le domaine visé en classes et en définissant les responsabilités de chacune ainsi que les collaborations entre classes. Le noyau de notre application VRP-MH est constitué d'un ensemble de classes permettant l'initialisation des problèmes de tournées de véhicules, leurs résolutions par différentes métaheuristiques, l'affichage des résultats ainsi que l'illustration des tournées obtenues sous forme graphique. Les classes de l'application VRP-MH sont présentées à travers le diagramme de classes de la *Figure 2.2*.

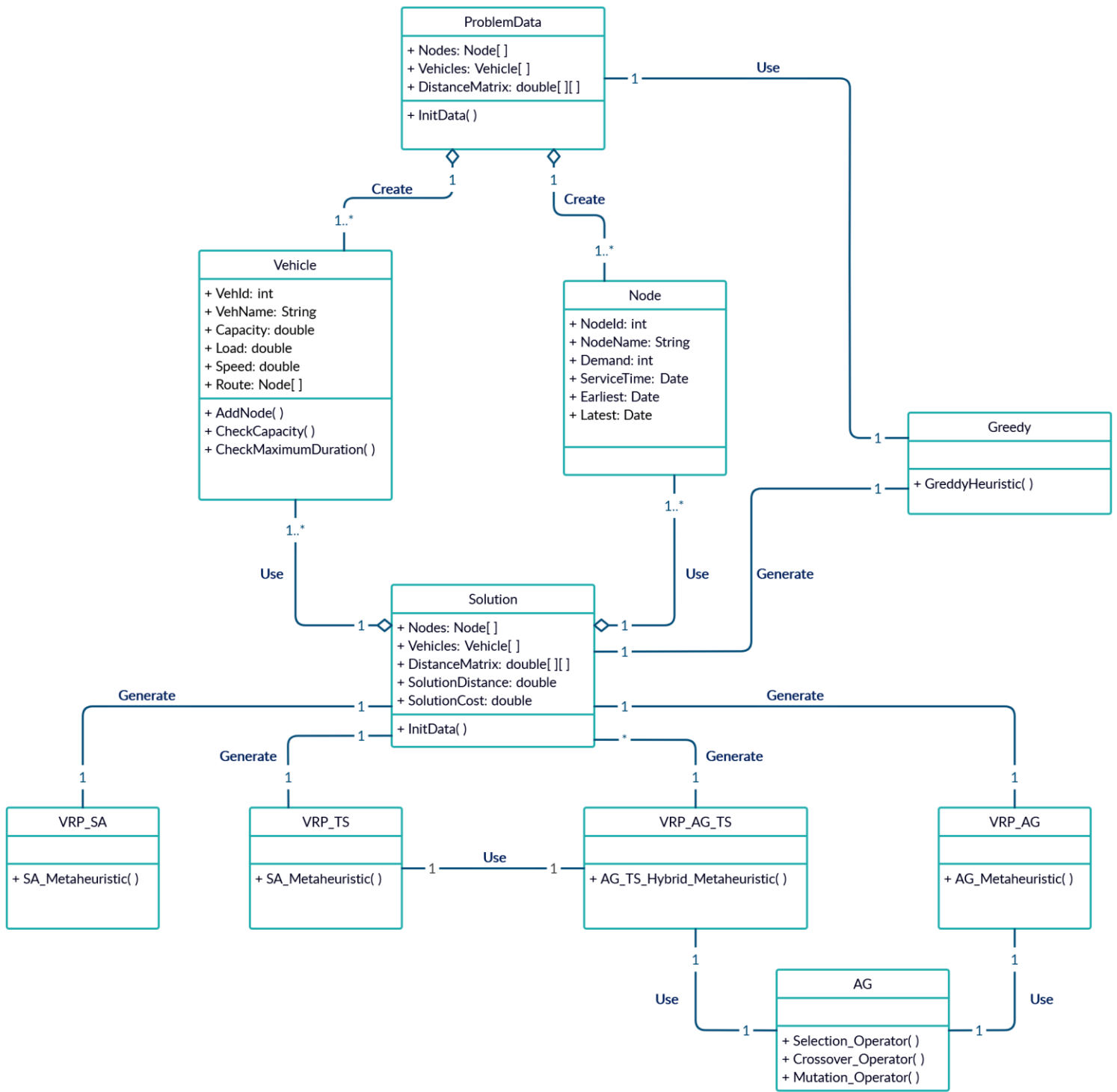


Figure 2.2 ; Diagramme de classes de l'application VRP-MH

Une brève description des classes de ce diagramme est donnée ci-dessous :

**Node** : c'est la classe qui représente les nœuds (clients ou point de collecte) du problème de tournée considéré.

**Vehicle** : c'est la classe qui représente les véhicules du problème de tournée considéré. Cette classe contient les méthodes suivantes :

- *AddNode* : permettant d'ajouter un nœud au circuit (route) du véhicule.
- *CheckCapacity* : avant l'ajout d'un nœud, cette méthode vérifie d'abord si la demande du nœud à ajouter ne dépasse pas la capacité du véhicule.
- *CheckWindowTime* : vérifie si les bornes de la fenêtre de temps sont respectées pour le nœud à ajouter.
- *CheckMaximumDuration* : avant l'ajout d'un nœud, cette méthode vérifie d'abord si la durée de la tournée ne dépasse pas la durée maximale autorisée.

**ProblemData** : classe contenant toutes les données nécessaires du problème, elle contient une méthode permettant l'initialisation de ces données.

**Solution** : représente une solution d'un problème de tournée, elle est composée de tous les nœuds et les véhicules du problème avec leurs routes respectives, chaque route passe par un ensemble de nœuds dans un ordre bien déterminé. Cette classe dispose aussi des données qui concernent la distance totale parcourue et le coût total de transport de la solution.

**Greedy** : cette classe utilise les données du problème obtenu par la classe *ProblemData* et génère une solution initiale en utilisant une heuristique gloutonne (voir).

**VRP\_SA** : cette classe exécute la métaheuristique du recuit simulé pour générer une solution au problème de tournée. La solution initiale par laquelle commence l'exécution de la métaheuristique est obtenue à partir de la classe *Greedy*.

**VRP\_TS** : cette classe exécute la métaheuristique de la recherche tabou pour générer une solution au problème de tournée. La solution initiale par laquelle commence l'exécution de la métaheuristique est obtenue à partir de la classe *Greedy*.

**AG** : classe contenant les méthodes des différents opérateurs génétiques (sélection, croisement, mutation) ainsi que la procédure de correction des chromosomes infaisables. Cette classe est essentielle pour les classe *VRP\_AG* et *VRP\_AG\_TS*.

**VRP\_AG** : cette classe utilise un algorithme génétique pour générer une solution au problème de tournée, elle fait appel à la classe *AG* pour exécuter les différents opérateurs génétiques. La population initiale de l'AG exécuté est composée d'un mélange de solutions initialisées aléatoirement et d'autres solutions obtenues à partir de la classe *Greedy*.

**VRP\_AG\_TS** : fait appel aux classes *AG* et *VRP\_TS* pour exécuter une métaheuristique hybride permettant d'hybrider un algorithme génétique avec recherche tabou. A l'instar de *VRP\_AG*, la population initiale de l'AG est composée d'un mélange de solutions initialisées aléatoirement et d'autres solutions obtenues à partir de la classe *Greedy*.

## **6.3. Description de l'environnement de développement**

### **6.3.1. Langage JAVA**

Java est un langage de programmation orienté objet développé par Sun Microsystems (aujourd'hui racheté par Oracle).

Une particularité de Java est que les logiciels écrits dans ce langage sont compilés vers une représentation binaire intermédiaire qui peut être exécutée dans une machine virtuelle Java (JVM) en faisant abstraction du système d'exploitation. Nous avons implémenté notre application VRP-MH avec ce langage en utilisant l'IDE Eclipse.

### 6.3.2. L'IDE Eclipse

**Définition d'un IDE :** Un IDE (environnement de développement intégré) est un outil qui a pour objectif de faciliter le développement sous un ensemble restreint de langages. Il contient un ensemble de fonctionnalités permettant de faciliter la tâche au programmeur. Les principales fonctionnalités d'un IDE sont :

- Un éditeur de texte intelligent avec coloration des mots clés.
- Un compilateur permettant d'exécuter rapidement le programme.
- Un débogueur permettant de traquer les erreurs du programme.

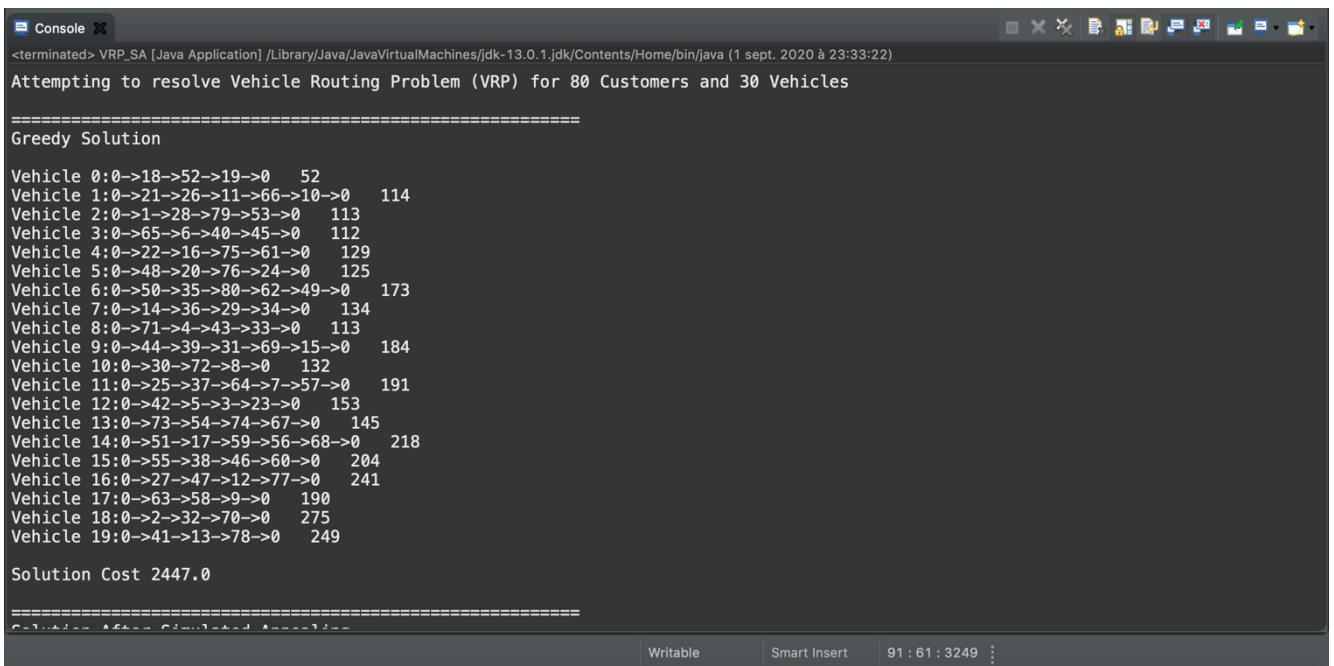
**Eclipse :** Est un IDE libre, extensible et polyvalent développé en JAVA. Il permet de créer des projets de développement sous n'importe quel langage de programmation. Eclipse est principalement développé autour de la notion de plug-in. Ceci permet à ses utilisateurs de l'adapter selon leurs besoins.

[<http://openclassrooms.com/courses/apprenez-a-programmer-en-java.>]

## 7. Démonstration de l'application VRP-MH

### 7.1. L'interface des résultats

Les deux *Figures 2.3 et 2.4* suivantes représentent un aperçu des résultats obtenu par notre application :



```
<terminated> VRP_SA [Java Application] /Library/Java/JavaVirtualMachines/jdk-13.0.1.jdk/Contents/Home/bin/java (1 sept. 2020 à 23:33:22)
Attempting to resolve Vehicle Routing Problem (VRP) for 80 Customers and 30 Vehicles
=====
Greedy Solution
Vehicle 0:0->18->52->19->0 52
Vehicle 1:0->21->26->11->66->10->0 114
Vehicle 2:0->1->28->79->53->0 113
Vehicle 3:0->65->6->40->45->0 112
Vehicle 4:0->22->16->75->61->0 129
Vehicle 5:0->48->20->76->24->0 125
Vehicle 6:0->50->35->80->62->49->0 173
Vehicle 7:0->14->36->29->34->0 134
Vehicle 8:0->71->4->43->33->0 113
Vehicle 9:0->44->39->31->69->15->0 184
Vehicle 10:0->30->72->8->0 132
Vehicle 11:0->25->37->64->7->57->0 191
Vehicle 12:0->42->5->3->23->0 153
Vehicle 13:0->73->54->74->67->0 145
Vehicle 14:0->51->17->59->56->68->0 218
Vehicle 15:0->55->38->46->60->0 204
Vehicle 16:0->27->47->12->77->0 241
Vehicle 17:0->63->58->9->0 190
Vehicle 18:0->2->32->70->0 275
Vehicle 19:0->41->13->78->0 249

Solution Cost 2447.0
=====
Solution After Simulated Annealing
```

Figure 2.3 : L'interface de résultats montrant la solution gloutonne après exécution du programme

```

Console
<terminated> VRP_AG [Java Application] /Library/Java/JavaVirtualMachines/jdk-13.0.1.jdk/Contents/Home/bin/java (1 sept. 2020 à 23:32:50)
=====
Best Solution After AG metaheuristic
Vehicle 0:0->18->52->19->0 52
Vehicle 1:0->21->26->11->66->10->0 114
Vehicle 2:0->1->28->79->53->0 113
Vehicle 3:0->65->6->40->45->0 112
Vehicle 4:0->22->16->75->61->0 129
Vehicle 5:0->48->20->76->24->0 125
Vehicle 6:0->50->35->80->62->49->0 173
Vehicle 7:0->14->36->29->34->0 134
Vehicle 8:0->71->4->43->33->0 113
Vehicle 9:0->44->39->31->69->15->0 184
Vehicle 10:0->30->72->8->0 132
Vehicle 11:0->25->37->64->7->57->0 191
Vehicle 12:0->42->5->3->23->0 153
Vehicle 13:0->73->54->74->67->0 145
Vehicle 14:0->51->17->59->56->68->0 218
Vehicle 15:0->55->38->46->60->0 204
Vehicle 16:0->27->47->77->0 169
Vehicle 17:0->63->58->9->0 190
Vehicle 18:0->2->12->32->0 212
Vehicle 19:0->70->41->13->78->0 263

Solution Cost 2326.0

Opération effectuée en: 4.191 secondes.
Writable Smart Insert 20 : 1 : 315

```

Figure 2.4 : L'interface de résultats montrant la solution de l'algorithme AG+TS

## 7.2. Exemple d'affichage des résultats par l'application VRP-MH

Les deux Figures 2.5 et 2.6 suivantes présentent deux illustrations de tournées obtenues par l'application VRP-MH

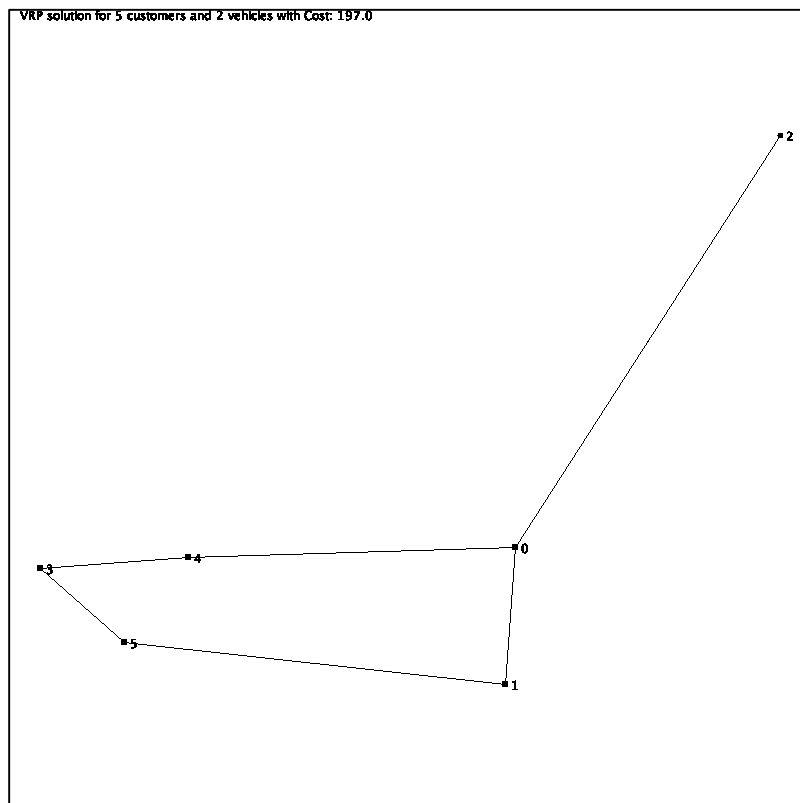


Figure 2.5 : Exemple d'une solution de l'algorithme RS

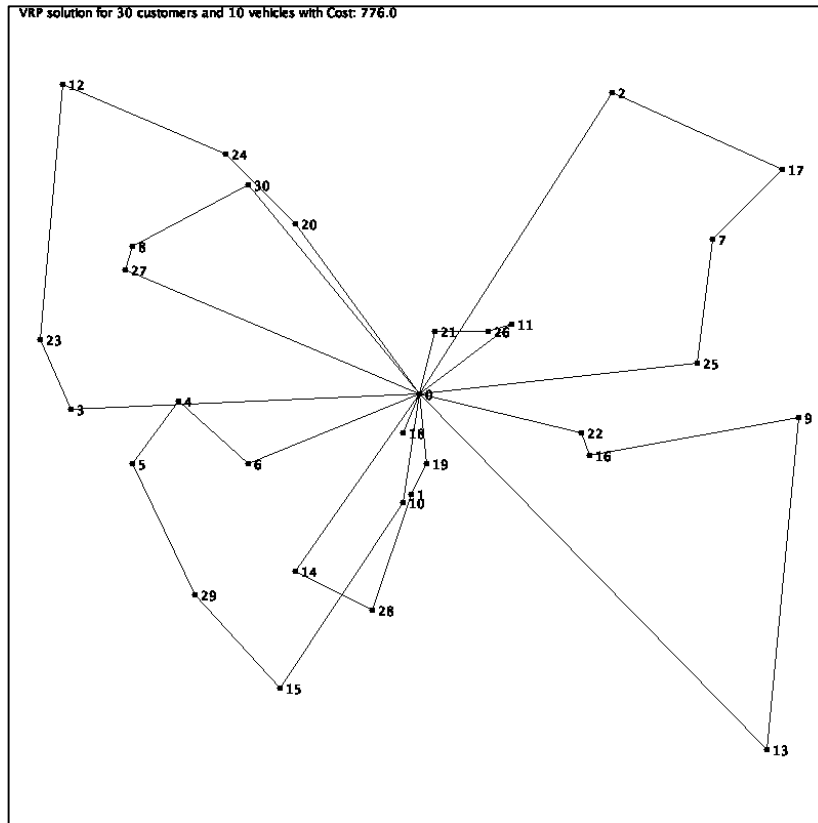


Figure 2.6 : Exemple d'une solution de l'algorithme AG+TS

### 7.3. Performances de l'application VRP-MH

Pour évaluer les performances et l'applicabilité de notre application VRP-MH, plusieurs expériences numériques ont été exécutées et les résultats les plus pertinents sont indiqués dans cette section. Notons que l'application VRP-MH a été exécutée sur un MacBook équipé d'un processeur Intel i5 (2.7 GHz), disposant d'une mémoire de 8Go et fonctionnant sous un système d'exploitation MacOS version 10.15.

#### 7.3.1. Jeux de donnée

Notre procédure de génération des différentes instances consiste tout d'abord à fixer le nombre de nœuds et de véhicules. Ensuite, de générer aléatoirement les coordonnées des nœuds sur une grille de  $100 \times 100$ . Nous calculons enfin les distances euclidiennes entre chaque paire de nœud.

Les différents paramètres utilisés pour la génération des instances de test sont présentés comme suit :

- Nombre de nœuds ( $N$ ) et nombre de véhicules ( $K$ ) : nous avons considéré des instances de différentes tailles avec 5, 8, 10, 15, 30, 50, 80 et 100 nœuds et respectivement 2, 3, 4, 6, 10, 15, 30, 35 véhicules ;



- Demandes des clients : générées aléatoirement suivant une loi uniforme dans l'intervalle  $[4, 10]$  ;
- La capacité des véhicules ( $C_i$ ): nous supposons que les capacités sont les mêmes pour tous les véhicules, dans ce cas tous les  $C_i$  sont remplacés par  $C$ . Pour nos tests, nous avons considéré deux capacités de véhicules :
  - $C1 = (\text{Somme totales des demandes} / \text{Nombre de véhicules}) + 10$  ;
  - $C2 = (\text{Somme totales des demandes} / \text{Nombre de véhicules}) + 30$  ;
- Distance entre les clients : la distance euclidienne entre les nœuds ayant des coordonnées choisis aléatoirement dans l'espace  $[0, 100] \times [0, 100]$  ;
- Temps de déplacement entre les clients : nous supposons que les durées de trajet entre les clients sont égale à la distance entre ces derniers (nous supposons alors que les véhicules ont tous la même vitesse égale à 1 u.d/u.t (u.d : unité de distance / u.t : unité de temps) ;
- Fenêtres de temps des clients : initialisées aléatoirement.
- Durée maximale des tournées (T) : nous avons considéré deux durées maximales des tournées :
  - T1 = durée infinie (sans contrainte de durée maximale de tournée)
  - T2 = (Durée de trajet maximale)  $\times$  2

Par ailleurs, les paramètres des différentes métaheuristiques proposées sont présentées comme suit :

### 7.3.2. Réglage des paramètres des métaheuristiques utilisées

Après une série de tests pour chaque métaheuristique, nous avons opté pour les paramètres suivants:

- Pour la métaheuristique RS :
  - Température initiale = 10 000 ;
  - Paramètre de refroidissement = 0.98 ;
  - Nombre d'itération à chaque température = 100 ;
- Pour la métaheuristique RT :
  - Nombre d'itérations maximal = 2000
  - Nombre de voisins par itération = 200 ;
  - La taille de la liste tabou (*TABU\_Horizon*) = 10 ;
- Pour la métaheuristique AG : voir le *Tableau 2.4*
- Pour la métaheuristique hybride AG+TS : paramètres de de l'AG et de la RT fusionnés.

	Chromosome	
	L'ordre dans lequel les nœuds sont visités	Le nombre de nœuds parcourus par chaque véhicule
Représentation	Nombres entiers	Nombres entiers
Type de croisement	Deux points	Un point
Probabilité de croisement	0.75	
Type de mutation	Standard	Standard
Probabilité de mutation	0.008	
Taille de la population	30	
Nombre d'itérations	1500	

*Tableau 2.3: Paramètres de l'AG adopté*

### 7.3.3. Résultats obtenus

Les résultats obtenus par l'application VRP-MH pour chaque instance de test sont détaillés à travers le *Tableau 2.5*. Ces résultats concernent les distances totales des tournées, le nombre de véhicules utilisés ainsi que le temps d'exécution obtenus pour chaque métaheuristique. Nous avons utilisé la méthode exacte Branch & Bound pour valider les résultats des 4 métaheuristicques implémentées dans l'application VRP-MH.

Notons bien qu'un délai de 1200 secondes (20 minutes) a été imposé sur l'algorithme B&B. Ainsi chaque fois que cette limite a été atteinte, nous interrompons l'exécution au niveau de Lingo et considérons que l'algorithme B&B n'a pas été en mesure de trouver une solution optimale dans un temps raisonnable.

Notons par ailleurs, que nous avons exécuté vingt (20) fois chaque métaheuristique (RS, RT, AG et AG+RT) et avons rapporté dans le *Tableau 2.5* les meilleurs résultats obtenus pour les différentes instances.

Instance	Clients (N)	Véhicules (K)	Capacités (C)	Durée max (T)	Greedy		Solveur Lingo			Recuit Simulé (RS)			Recherche Tabou (RT)			Algorithme Génétique (AG)			Hybridation AG+RT (10 it)		
					Distance obtenue (Km)	Nombre de véhicules	Distance obtenue (Km)	Nombre de véhicules	Temps d'exécution (s)	Distance obtenue (Km)	Nombre de véhicules	Temps d'exécution (s)	Distance obtenue (Km)	Nombre de véhicules	Temps d'exécution (s)	Distance obtenue (Km)	Nombre de véhicules	Temps d'exécution (s)	Distance obtenue (Km)	Nombre de véhicules	Temps d'exécution (s)
1	5	2	C1	T1	203	2	<b>197</b>	2	1s	<b>197</b>	2	0.125	<b>197</b>	2	0.04s	<b>197</b>	2	0.085	<b>197</b>	2	0,444s
2				T2	203	2	<b>197</b>	2	1s	<b>197</b>	2	0.122	<b>197</b>	2	0.05	<b>197</b>	2	0.09	<b>197</b>	2	0,654
3			C2	T1	193	1	<b>189</b>	1	1s	<b>189</b>	1	0.159	<b>189</b>	1	0.059	<b>189</b>	1	0.088	<b>189</b>	1	0.465
4				T2	193	1	<b>189</b>	1	1s	<b>189</b>	1	0.151	<b>189</b>	1	0.038	<b>189</b>	1	0.079	<b>189</b>	1	0.561
5	8	3	C1	T1	285.0	2	<b>278</b>	2	8s	<b>278.0</b>	2	0.128	<b>278.0</b>	2	0.048	<b>278.0</b>	2	0.129	<b>278.0</b>	2	0.819
6				T2	308.0	3	<b>286</b>		10s	<b>286.0</b>	3	0.145	<b>286.0</b>	3	0.04	<b>286.0</b>	3	0.632	<b>286.0</b>	3	0.632
7			C2	T1	286.0	2	<b>243</b>	2	8s	<b>243.0</b>	2	0.146	<b>243.0</b>	2	0.054	<b>243.0</b>	2	0.16	<b>243.0</b>	2	1.016
8				T2	308.0	3	<b>254</b>	2	6s	<b>254.0</b>	2	0.149	<b>254.0</b>	2	0.088	<b>254.0</b>	2	0.154	<b>254.0</b>	2	0.89
9	10	4	C1	T1	403.0	3	<b>318</b>	3	20s	<b>318.0</b>	3	1.451	<b>318.0</b>	3	0.044	<b>318.0</b>	3	0.187	<b>318.0</b>	3	1.152
10				T2	414.0	4	<b>318</b>	3	18s	<b>318.0</b>	3	0.125	<b>318.0</b>	3	0.054	<b>318.0</b>	3	0.181	<b>318.0</b>	3	1.261
11			C2	T1	295.0	2	<b>275</b>	2	22s	<b>275.0</b>	2	0.166	<b>275.0</b>	2	0.056	<b>275.0</b>	2	0.148	<b>275.0</b>	2	1.414
12				T2	383.0	3	<b>292</b>	3	20s	<b>292.0</b>	3	0.203	<b>292.0</b>	3	0.049	<b>292.0</b>	3	0.167	<b>292.0</b>	3	1.342
13	15	6	C1	T1	589.0	4	<b>545</b>	4	< 20min	<b>545.0</b>	4	0.135	<b>545.0</b>	4	0.053	<b>545.0</b>	4	0.234	<b>545.0</b>	4	2.174
14				T2	589.0	4	<b>545</b>	3	< 20min	<b>545.0</b>	3	0.326	<b>545.0</b>	3	0.061	<b>545.0</b>	3	0.287	<b>545.0</b>	4	1.807
15			C2	T1	593.0	3	<b>452</b>	3	< 20min	<b>452.0</b>	3	0.14	<b>452.0</b>	3	0.069	471.0	3	0.292	<b>452.0</b>	3	3.016
16				T2	583.0	4	<b>447</b>	3	< 20min	<b>447.0</b>	3	0.793	476.0	3	0.354	489.0	3	0.259	476.0	3	2.859
17	30	10	C1	T1	975.0	7	N / A	N / A	N / A	856.0	7	0.174	856.0	7	0.064	817.0	8	0.574	<b>776.0</b>	8	3.83
18				T2	948.0	8	N / A	N / A	N / A	<b>779.0</b>	7	0.061	793.0	7	0.164	784.0	8	0.537	781.0	8	4.287
19			C2	T1	766.0	5	N / A	N / A	N / A	623.0	5	0.124	631.0	5	0.076	627.0	5	0.606	<b>620.0</b>	5	6.068

20				T2	845.0	6	N / A	N / A	N / A	<b><u>629.0</u></b>	5	0.195	637.0	5	0.085	634.0	5	0.703	634.0	5	6.216
21	50	15	C1	T1	1436.0	11	N / A	N / A	N / A	1205.0	11	0.186	1207.0	11	0.065	1162.0	11	0.988	<b><u>1158.0</u></b>	8	7.402
22				T2	1526.0	12	N / A	N / A	N / A	1181.0	11	0.238	1176.0	11	0.061	1162.0	11	1.109	<b><u>1152.0</u></b>	11	7.509
23			C2	T1	1096.0	7	N / A	N / A	N / A	910.0	7	0.131	921.0	7	0.069	940.0	7	1.174	<b><u>907.0</u></b>	7	12.77
24				T2	1177.0	7	N / A	N / A	N / A	910.0	7	0.28	929.0	7	0.067	936.0	8	1.162	<b><u>892.0</u></b>	7	12.02
25	80	30	C1	T1	2447.0	20	N / A	N / A	N / A	2204.0	20	0.191	2192.0	20	0.057	2321.0	20	3.76	<b><u>2099.0</u></b>	21	14.87
26				T2	2511.0	21	N / A	N / A	N / A	2165.0	20	0.232	2114.0	20	0.06	2353.0	20	1.478	<b><u>2064.0</u></b>	21	14.19
27			C2	T1	1699.0	12	N / A	N / A	N / A	<b><u>1459.0</u></b>	12	0.172	1494.0	12	0.077	1616.0	12	5.822	1534.0	12	11.19
28				T2	1761.0	13	N / A	N / A	N / A	1452.0	12	0.42	<b><u>1448.0</u></b>	12	0.085	1677.0	12	6.905	1537.0	13	9.557
29	100	35	C1	T1	2902.0	24	N / A	N / A	N / A	2475.0	24	0.167	2476.0	24	0.062	2667.0	24	6.351	<b><u>2418.0</u></b>	24	13.332
30				T2	2588.0	24	N / A	N / A	N / A	2457.0	24	0.367	<b><u>2387.0</u></b>	24	0.065	2504.0	24	1.583	2402.0	24	3.751
31			C2	T1	2049.0	14	N / A	N / A	N / A	1923.0	14	0.166	1804.0	14	0.08	1907.0	14	8.103	<b><u>1787.0</u></b>	15	20.106
32				T2	2043.0	16	N / A	N / A	N / A	<b><u>1638.0</u></b>	14	0.297	1677.0	15	0.085	1967.0	16	2.839	1705.0	15	18.196

Tableau 2.4 : Résultats obtenus par l'application VRP-MH

\*N / A : L'algorithme Branch & Bound n'a pas été en mesure de trouver une solution optimale dans un temps raisonnable.

Les résultats montrent que le coût (distance totale parcouru) augmente avec le nombre de clients. Comme on pouvait s'y attendre, nous constatons aussi, que pour chaque instance, l'augmentation des capacités des véhicules aura pour conséquence la satisfaction de plus de clients pour chaque circuit. Ceci permettra d'élaborer des tournées plus efficaces et plus optimisées et ainsi de réduire les coûts. Lorsque nous ajoutons la contrainte de la durée maximale  $T_2$ , on remarque une légère amélioration des résultats parce que en limitant la durée, le nombre de client dans le circuit sera aussi limité et toutes les solutions contenant plusieurs clients seront éliminer, donc l'algorithme vas calculer plus efficacement en négligeant les longs circuits et arrivera à la meilleure solution rapidement.

#### **7.3.4. La validation de résultats de VRP-MH par la méthode exacte Branch & Bound :**

Nous remarquons clairement à travers le *Tableau 2.5* que l'algorithme Branch & Bound a été en mesure de résoudre toutes les instances de test jusqu'à 15 nœuds (clients). De plus, à mesure que l'ampleur des problèmes (instances) augmente, les temps d'exécution de Branch & Bound augmentent considérablement d'une manière exponentielle. A partir des instances moyennes de 30 clients et plus, l'algorithme B&B ne peut même pas fournir de solutions réalisables.

Par ailleurs, nous remarquons pour les petites instances (moins de 30 clients) que les quatre métaheuristiques donnent (à quelques exceptions) des résultats égaux à ceux de la méthode exacte B&B avec évidemment des temps d'exécution largement inférieurs.

La fiabilité et la validité des quatre métaheuristiques ont été prouvées à travers cette comparaison, nous pouvons donc utiliser l'application VRP-MH pour les moyennes et grandes instances.

## Conclusion générale et perspectives

L'objectif de ce travail de master était de développer une application permettant l'optimisation des problèmes de tournées de véhicules intégrant les contraintes temporelles de fenêtres de temps des clients/points de collecte et de durée maximale des tournées. L'application développée baptisée VRP-MH (Vehicle Routing Problem – MetaHeuristics) utilise quatre approches basées sur les métaheuristiques pour la résolution des problèmes de tournée. Les métaheuristiques implémentées au sein de l'application VRP-MH sont les suivantes : Recuit simulé, Recherche tabou, Algorithme génétique et une dernière approche hybridant un algorithme génétique avec la recherche tabou.

Dans la première partie de ce mémoire, nous avons évoqué le problème VRP et ses principales variantes ainsi qu'une synthèse des méthodes de résolution des problèmes de type VRP. Une attention particulière a été accordée aux méthodes de résolution basées sur les métaheuristiques déployées dans l'application VRP-MH.

La deuxième partie concernait la conception et le développement de l'application VRP-MH. L'analyse des besoins et la conception sont faites en utilisant le langage de modélisation UML. La modélisation de notre application a débuté par la description des cas d'utilisation, qui ont permis d'aboutir au diagramme de classes représentant la structure générale de notre application. Dans cette partie, nous avons également présenté une démonstration de l'exécution de l'application VRP-MH en présentant et interprétant les résultats obtenus.

Les perspectives futures de ce travail concernent notamment l'extension de l'application VRP-MH à travers notamment le développement d'autres métaheuristiques hybrides permettant de résoudre efficacement les problèmes de tournées de véhicules.

## Références

- Open source platform for developers “GitHub” : <https://github.com/nimich/VehicleRouting>.
- Site Web de formation en ligne “OpenClassrooms” : <https://openclassrooms.com/fr/>
- K. Q. Zhu. « A new genetic algorithm for VRPTW ». International Conference on Artificial Intelligence, Las Vegas, USA, April 13 2000.
- B. M. Baker and M.A. Ayechev . « A genetic algorithm for the vehicle routing problem ». Computers and Operations Research 30, pages 787–800, 2003.
- Boussaid, I. (2013). Perfectionnement de métaheuristiques pour l'optimisation continue (Doctoral dissertation, Paris Est).
- Benaichouche, A. N. (2014). Conception de métaheuristiques d'optimisation pour la segmentation d'images: application aux images IRM du cerveau et aux images de tomographie par émission de positons (Doctoral dissertation).
- Widmer, M. (2001, April). Les métaheuristiques: des outils performants pour les problèmes industriels. In 3ème Conférence Francophone de Modélisation et Simulation MOSIM (Vol. 1, pp. 25-27).
- Mekamcha ,K.(2019). Réalisation d’un support d’aide à la décision pour la gestion du problème stochastique de la collecte des déchets ménagers dans une zone urbaine.
- Sevaux, M. (2004). Métaheuristiques: Stratégies pour l'optimisation de la production de biens et de services (Doctoral dissertation).
- Vidal, T., Crainic, T., Gendreau, M., & Prins, C. (2011). Heuristiques pour les problèmes de tournées de véhicules multi-attributs. CIRRELT.
- M. H Hugos 2003. Journal of business logistics, 22(2):1–25, 2001
- Applegate, D. L., Bixby, R. E., Chvatal, V., & Cook, W. J. (2006). The traveling salesman problem: a computational study. Princeton university press.
- Bernard, T. T., & Pauline, F. L. Heuristiques du problème du voyageur de commerce.
- Rizet, C., & Keita, B. (2005). Chaînes logistiques et consommation d'énergie: cas du yaourt et du jean.

- KHELFI, M. M. F. (2017). Proposition de solutions pour l'optimisation des chaînes logistiques (Doctoral dissertation, Université d'Oran).
- Guerra, L., Murino, T., & Romano, E. (2007). The location-routing problem: an innovative approach. 6th WSEAS Transactions on System Science and Simulation in Engineering, Venice, Italy, 21-23.