RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohammed Seddik BENYAHIA -Jijel-

Faculté des Sciences Exactes et de l'Informatique

Département d'Informatique

Mémoire

Pour l'obtention du diplôme **MASTER**

EN : INFORMATIQUE

Spécialité : Systèmes d'Information et Aide à la Décision

Thème

Contribution à la résolution coopérative approchée du Problème de tournées de véhicules

Réalisé par :

MOHAMEDI DJAFER

LADJEROUD AHMED

Soutenue publiquement, le 23/06/2016, devant le jury composé de :

M^me KERRADA WIDAD	Présidente
M^r LATER AZZEDINE	Examinateur
M^me MERYEM BERGHIDA	Encadreur

Année Universitaire : 2015-2016

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohammed Seddik BENYAHIA -Jijel-

Faculté des Sciences Exactes et de l'Informatique

Département d'Informatique

Mémoire

Pour l'obtention du diplôme **MASTER**

EN : INFORMATIQUE

Spécialité : Systèmes d'Information et Aide à la Décision

Thème

Contribution à la résolution coopérative approchée du Problème de tournées de véhicules

Réalisé par :

MOHAMEDI DJAFER

LADJEROUD AHMED

Soutenue publiquement, le 23/06/2016, devant le jury composé de :

M^me KERRADA WIDAD	Présidente
M^r LATER AZZEDINE	Examinateur
M^me MERYEM BERGHIDA	Encadreur

Année Universitaire : 2015-2016

Dédicace

Je dédie ce mémoire à mes chers parents à ma mère ***Nabila*** et mon père ***Mohammed*** sources de mes joies, secrets de ma force vous serez toujours le modèle qui me guide dans la vie : Papa, dans ta détermination, ta force et ton honnêteté, Maman dans ta bonté, ta patience et ton dévouement pour nous tous. Merci pour tous vos sacrifices pour que vos enfants grandissent et prospèrent, Merci de lutter sans relâche, malgré les péripéties de la vie au bien être de vos enfants, Merci pour tous vous encouragements à continuer mes études, Merci d'être tout simplement mes parents, C'est à vous que je dois cette réussite et je suis fière de vous l'offrir.

À mes frères : ***Walid*** et ***Anes*** en témoignage de mes sentiments de fraternité et d'amour que je porte pour vous. Je vous dédie ce travail avec tous mes vœux de bonheur, santé et un grand avenir plein de joie et de réussite.

À la mémoire de mes Grands-Parents paternels et mon frère ***Zakaria***. Le destin ne m'a pas donné la chance de vous connaitre et de vous exprimer tout mon amour et mon respect. Puisse Dieu le tout puissant vous accorder sa clémence, et vous accueillir dans son saint paradis ***Firdaws inchallah.***

À mes Grand Parents Maternels. Quoique je puisse dire, je ne peux exprimer mes sentiments d'amour et de respect à votre égard. Puisse le tout puissant, vous procurer une longue et heureuse vie.

À tous ma famille, mes amis, mes camarades, et toutes les personnes rares qui font partie de ma vie, ***Ahmed***, ***Chouaib, Zaid, Nadji, Amine.Madi, Fouad, Mohsen, Oussama, Amine.Abdellah,*** sans oublier mon professeur et ma sœur ***Birouk.Wafa.***

Et à toutes les personnes que je n'ai pas pu citer.

Mohamedi Djafer

Dédicace

Je dédie ce mémoire à mes chers parents à ma mère ***Hadjira*** et mon père ***Said*** sources de mes joies, secrets de ma force vous serez toujours le modèle qui me guide dans la vie : Papa, dans ta détermination, ta force et ton honnêteté, Maman dans ta bonté, ta patience et ton dévouement pour nous tous. Merci pour tous vos sacrifices pour que vos enfants grandissent et prospèrent, Merci de lutter sans relâche, malgré les péripéties de la vie au bien être de vos enfants, Merci pour tous vous encouragements à continuer mes études, Merci d'être tout simplement mes parents, C'est à vous que je dois cette réussite et je suis fière de vous l'offrir.

A ma sœur : ***Madjda*** en témoignage de l'attachement, de l'amour que je te porte. Je te dédie ce travail avec tous mes vœux de bonheur, santé et de réussite.

A Mon frère, le bijou de la famille : ***Soufiane*** Je te souhaite un avenir plein de joie, de bonheur, de réussite et de sérénité. Je t'exprime à travers ce travail mes sentiments de fraternité et d'amour.

À la mémoire de mes Grands-Parents paternels et mon grand-père Maternels. Le destin ne nous a pas laissé le temps pour jouir ce bonheur ensemble et de vous exprimer tout mon respect. Puisse Dieu le tout puissant vous accorder sa clémence, et vous accueillir dans son saint paradis ***Firdaws inchallah.***

À ma Grand-mère Maternels. Quoique je puisse dire, je ne peux t'exprimer mes sentiments d'amour et de respect à votre égard. Puisse le tout puissant, te procurer une longue et heureuse vie.

À tous ma famille, mes amis, mes camarades, et toutes les personnes rares qui font partie de ma vie, ***Djafer, Islam, Bassem, Riad, Zaid.***

Et à toutes les personnes que je n'ai pas pu citer.

Ladjeroud Ahmed

Remerciements

En tout premier lieu, on tient à remercier notre dieu Allah le tout puissant, de nous avoir donné la force pour survivre, l'audace pour dépasser toutes les difficultés et ainsi que la volonté et le courage pour la réalisation de ce travail.

On tient à exprimer notre profonde gratitude à notre encadreur, Docteur Berghida Meryem pour nous avoir encadré, orienté et guidé avec ses précieux conseils afin d'aboutir à la réalisation de ce travail.

On tient aussi à remercier les membres des jurys qui ont acceptés de juger et d'évaluer notre travail.

On adresse aussi nos vifs remerciements à nos familles pour leur encouragement et le soutien affectif et matériel qu'ils nous ont apporté tout au long de notre existence.

On remercie aussi nos enseignants, nos amis, nos camarades et toutes les personnes qui, de près ou de loin nous ont aidés pour l'achèvement de ce travail.

Table des matières

Liste des figures

Liste des tableaux

Liste des algorithmes

Introduction

Dans l'environnement concurrentiel d'aujourd'hui, il est évident que les entreprises doivent prendre des décisions stratégiques et opérationnelles afin d'optimiser et de gérer efficacement les processus dans leur système logistique.

La chaîne logistique implique de nombreuses activités telles que l'approvisionnement et la production. Pendant longtemps, la recherche sur la logistique a mis l'accent sur l'optimisation de ces activités depuis cette optimisation a éliminé la perte de temps. L'une des plus importantes décisions opérationnelles consiste à trouver des voies de véhicules optimales car il offre un grand potentiel pour réduire les coûts et améliorer la qualité de service.

En raison de cette importance économique et la priorité, les chercheurs ont accordé un grand intérêt aux problèmes de routages des véhicules, les entreprises sont amenées à chercher des moyens d'optimisation pour leurs véhicules sur le terrain qui soient capables de gérer un grand nombre de contraintes et de clients d'une manière efficace. Cette problématique est dérivé du problème de voyageur de commerce qui a évolué avec le temps pour faire apparaitre un nouveau problème connu actuellement sous le nom de problème de tournées de véhicules traduit de l'anglais Vehicle Routing Problem (VRP).

La réduction des coûts de transport a pris une grande intention par les développeurs qui ont proposé de nombreux logiciels sur le marché de plus en plus performants pour optimiser les tournées de véhicules. Cette problématique, a également fait l'objet de nombreux travaux académiques et a reçu une grande intention dans la littérature. Depuis plus de 50 ans, la recherche des méthodes de résolution plus efficace n'a cessé.

La formule la plus simple du VRP implique la construction des trajectoires des véhicules de livraison de manière à minimiser le coût, pour satisfaire la demande des clients répartis sur différentes zones géographiques, tout en respectant les contraintes de capacité des véhicules. D'autre part, des contraintes supplémentaires apparaissent telles que le temps réglementaire du travail des chauffeurs, les fenêtres de temps que certains clients...etc. Ce qui rend la résolution de ce problème plus complexe.

Le thème abordé dans le cadre de ce mémoire est la contribution à la résolution coopérative approchée du problème de tournées de véhicules. La problématique consiste à la mise en place d'un processus de recherche des solutions permettant de fournir une

planification de tournée d'une flotte de véhicules qui doit répondre aux demandes de livraisons d'un ensemble des clients.

La méthode proposée afin de parvenir à la résolution de ce problème, est une approche de coopération. Le concept de coopération consiste à faire coopérer plusieurs métaheuristiques esclaves dont le fonctionnement est supervisé par une métaheuristique dite maitre dans le but de profiter des avantages offerts par chacune des métaheuristiques esclaves.

Ce mémoire est organisé comme suit : Le premier chapitre présente une description du problème de tournées de véhicules ainsi que certains de ces variantes. Le chapitre suivant décrit l'approche proposée ainsi que les différentes métaheuristiques utilisées pour la résolution du problème de tournées de véhicules avec contrainte de capacité (CVRP). Le troisième chapitre expose les résultats auxquels est arrivée l'approche utilisée. Le dernier chapitre présente une récapitulation des différentes parties de ce travail ainsi que les perspectives attendues de ce dernier.

1. Chapitre 01 : Problème de tournées de véhicules (VRP)

1.1 Introduction

Le problème de tournées de véhicules (VRP) est l'un des problèmes d'optimisations les plus étudiés, depuis sa mise en œuvre par Dantzig et Ramser il y a plus de 50 ans sous le titre de « The Truck Dispatching Problem » [Dantzig 1959], des centaines d'articles ont été consacrées à la résolution exacte ou approximative des nombreuses variantes de ce problème, comme le VRP avec contrainte de capacité (CVRP), ou bien le VRP avec fenêtre de temps (VRPTW).

Ce chapitre va être consacré à la définition du problème de VRP et de quelques variantes de ce problème ainsi que quelques travaux récents qui les résout.

1.2 VRP : Vehicle Routing Problem

Le problème de tournées de véhicules noté VRP (Vehicle Routing Problem) est une classe de problèmes de recherche opérationnelle et d'optimisation qui a reçu beaucoup d'attention dans la littérature.

Le VRP consiste à déterminer pour un ensemble de véhicules, les itinéraires à prendre pour servir un ensemble de clients représentés par des points géographiques que ce soit pour intervention, visite ou bien pour une collecte/livraison à temps fini, L'objectif lié à ce problème est de minimiser les coûts de livraison tel que la distance parcourue.

Le VRP est une extension du problème de voyageur de commerce, donc il appartient à la classe des problèmes NP-difficile [Laporte 1992].

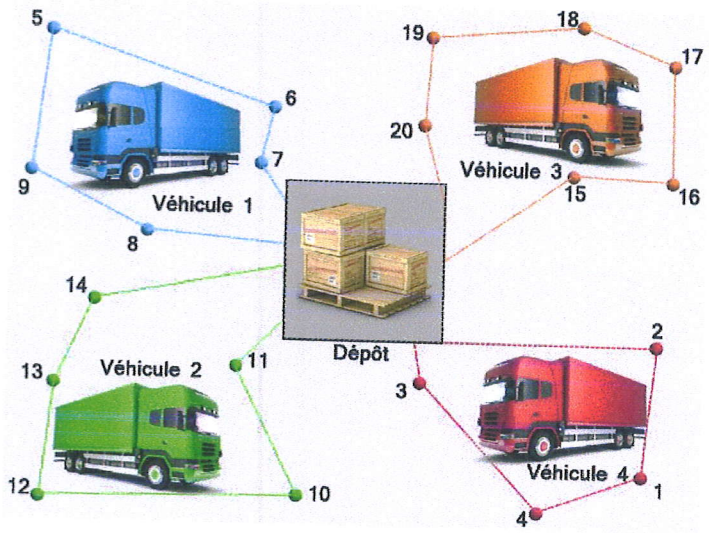


Figure 1.1 : Représentation d'une solution VRP

1.3 Variantes de VRP

1.3.1 CVRP : Capacitated Vehicle Routing Problem

CVRP est un problème de tournées de véhicules formulé par [Christofides 1976] dans lequel une flotte fixée de véhicules de livraison avec une capacité uniforme (similaire) doit servir les exigences des clients connus pour un seul produit dans un dépôt commun et avec un coût de transit minimum.

Le CVRP est la version de base de VRP où une seule contrainte peut s'imposer : chaque véhicule doit avoir une capacité prédéfinie. Dans ce qui suit nous présentons la formulation mathématique de CVRP proposée par Rego et Roucairol [Rego 1994] :

1.3.1.1 Formulation mathématique du CVRP

Soit $G = (N, A)$ un graphe où $N = \{1, \dots, n\}$ est un ensemble des sommets avec le sommet 1 fixé comme dépôt et $A = \{(i,j) | (i,j) \in N^2 et\ i \neq j\}$ est l'ensemble des arcs. $N' = N \backslash \{1\}$ est l'ensemble des clients à satisfaire obtenu par le fait d'enlever le dépôt de l'ensemble des sommets.

A chaque arc est associé un coût non négatif c_{ij} qui peut représenter le coût du voyage ou la durée du voyage entre i et j. Nous envisageons le cas où une flotte de m véhicules ayant la même capacité de transport D (véhicules homogènes) est disponible.

Les contraintes prises en compte dans cette formulation sont :

La contrainte de capacité : A chaque sommet i de N' est associé un poids d_i non négatif représentant la demande. La somme des poids d'une tournée ne doit pas dépasser la capacité D du véhicule.

La contrainte de temps total : Le temps total d'une tournée ne doit pas dépasser une borne T. Ce temps est la somme des temps des voyages entre les sommets et les temps d'arrêt à ces derniers.

La formulation de Rego [Rego 1994] adopte les notations suivantes :

Les constantes de données :

n = Le nombre de clients (ou sommets),

m = Le nombre de véhicules,

D_k = La capacité du véhicule k,

T_k = Temps maximal de la tournée du véhicule k,

d_i = La demande du sommet i,$(d_1 = 0)$,

t_i^k = Temps nécessaire au véhicule k pour charger ou décharger au sommet i,

t_{ij}^k = Temps nécessaire au véhicule k pour voyager du sommet i au sommet j,

c_{ij} = Coût ou distance du voyage du sommet i au sommet j,

N = Ensemble des sommets,

X = Matrice d'éléments $x_{ij} = \sum_{k=1}^{m} x_{ij}^k$,

Les variables de décision

$$x_{ij}^k = \begin{cases} 1 \; si \text{ le véhicule k voyage du sommet i au sommet j} \\ 0 \; sinon \end{cases} \quad (1.1)$$

Avec $X^k = (x_{ij}^k)$,

Fonction objectif à optimiser :

$$\textbf{\textit{Minimiser}}\; z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_i^j \sum_{k=1}^{n} x_{ij}^k \quad (1.2)$$

Sous :

$$\sum_{i=1}^{n}\sum_{k=1}^{m} x_{ij}^k = 1 \text{ , j} = 2, \dots, \text{n} \quad (1.3)$$

$$\sum_{j=1}^{n}\sum_{k=1}^{m} x_{ij}^k = 1 \text{ , i} = 2, \dots, \text{n} \quad (1.4)$$

$$\sum_{i=1}^{n} x_{ip}^k - \sum_{j=1}^{n} x_{pj}^k = 0, k = 1, \dots, m; p = 1, \dots, n \quad (1.5)$$

$$\sum_{i=1}^{n} d_i \left(\sum_{j=1}^{n} x_{ij}^k \right) \leq D_k, \text{k} = 1, \dots, \text{m} \quad (1.6)$$

$$\sum_{i=1}^{n} t_i^k \sum_{j=1}^{n} x_{ij}^k + \sum_{i=1}^{n}\sum_{i=j}^{n} t_{ij} x_{ij}^k \leq T_k, k = 1, \dots, m \quad (1.7)$$

$$\sum_{j=2}^{n} x_{1j}^k \leq 1, k = 1, \dots, m \quad (1.8)$$

$$\sum_{i=2}^{n} x_{i1}^{k} \leq 1, k = 1, \ldots, m \tag{1.9}$$

$$X^{k} \in S \tag{1.10}$$

Où S est donné par l'expression (1.11) [Ralphs 2003]

$$S = \left((x_{ij})^{k} \middle| \sum_{i \in Q} \sum_{j \notin Q} x_{ij}^{k} \geq 1, \forall Q \subset N, |Q| \geq 2; k = 1, \ldots, m \right) \tag{1.11}$$

Où Q est un ensemble de sommets visités par un seul véhicule.

La fonction objective (1.2) consiste à minimiser le coût total de transport. Les équations (1.3) et (1.4) assurent que chaque sommet ne soit servi qu'une seule fois par un et un seul véhicule. L'équation (1.5) assure la continuité d'une tournée par un véhicule : le sommet visité doit impérativement être quitté l'équation (1.6) assure le respect de la contrainte de capacité du véhicule. L'équation (1.7) assure le respect de la contrainte de la durée totale d'une tournée. Les équations (1.8) et (1.9) assurent le non dépassement de la disponibilité d'un véhicule. Un véhicule ne sort du dépôt et n'y revient qu'une seule fois. Finalement les équations (1.10), (1.11) assurent le respect des contraintes d'élimination des sous tournées (tournées revenant au client et pas au dépôt). D'une façon générale, le problème CVRP consiste à affecter chaque client à une tournée exécutée par un seul véhicule de capacité fixe. Ce véhicule commence et termine sa tournée au dépôt [Ralphs 2003].

1.3.2 VRPTW: Vehicle Routing Problem with Time Windows

Le VRPTW est similaire au problème de CVRP avec une restriction supplémentaire. Dans le VRPTW, chaque client doit être servi dans un intervalle de temps durant lequel il est disponible pour être visité [Cordeau 2001].

1.3.2.1 Formulation mathématique du VRPTW

Le VRPTW est une extension de CVRP dans lequel les contraintes de capacité sont imposées et que chaque client i est associé à un intervalle de temps $[a_i b_i]$ durant laquelle il peut être servi en poursuivant la formulation du CVRP d'autre variable de décision sont définies [Sahbi 2011] :

Pour $0 \leq i \leq net\ 1 \leq k \leq m, s_{ik}$ désigne l'instant où le véhicule k commence à servir le client i. Il en résulte deux contraintes supplémentaires :

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \qquad (1.12)$$

$$a_i \leq s_{ik} \leq b_i \qquad (1.13)$$

L'équation (1.12) permet de prendre en considération la durée de trajet entre deux clients consécutifs $i\ et\ j(t_{ij})$. Et l'équation (1.13) impose que le temps de début de livraison au client i soit compris dans la fenêtre du temps de ce dernier.

1.3.3 VRPB: Vehicle Routing Problem with Backhauls

Le VRPB est une extension de CVRP dans lequel l'ensemble des clients est divisée en deux sous-ensembles. Le premier sous-ensemble contient les clients de livraison, chacun nécessitant une certaine quantité de produits à délivrer. Le deuxième sous-ensemble contient les clients de collecte où une quantité donnée de produits doit être ramassée [Toth 2001].
Dans le VRPB, une contrainte de priorité entre les clients de collecte et de livraison est imposée : dès qu'une tournée sert les deux types de client, tous les clients de livraison doivent être servis avant les clients de collecte [Toth 2001].

1.3.3.1 Formulation Mathématique du VRPB

Soit :

n = Le nombre de clients de livraison,

m = Le nombre de clients de collecte,

M = Le nombre de véhicules,

Q_k = La capacité du véhicule k,

f_i = La demande à ramasser chez le client i ,

d_i = La de à livrer chez le client i,

c_{ij}^k = Le coût de déplacement entre les sommets i et j en utilisant le véhicule k (distance ou temps de parcours).

Soit les variables de décision du problème :

$$x_{ij}^k = \begin{cases} 1\ si\ (i,j)\ est\ parcourue\ par\ le\ véhicule\ k \\ 0\ sinon \end{cases}$$

Fonction objectif :

$$\textbf{Minimiser } z = \sum_{i=0}^{n+m}\sum_{j=0}^{n+m}\sum_{k=1}^{M} c_{ij}^{k} x_{ij}^{k} \qquad (1.14)$$

Sous les contraints suivantes :

$$\sum_{j=0}^{n+m}\sum_{k=1}^{M} x_{ij}^{k} = 1, \quad i = 1, \dots, n+m i \neq j \qquad (1.15)$$

$$\sum_{i=0}^{n+m}\sum_{k=1}^{M} x_{ij}^{k} = 1, \quad j = 1, \dots, n+m i \neq j \qquad (1.16)$$

$$\sum_{j=0}^{n+m}\sum_{k=1}^{M} x_{0j}^{k} = M \qquad (1.17)$$

$$\sum_{i=0}^{n+m}\sum_{k=1}^{M} x_{i0}^{k} = M \qquad (1.18)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n+m} d_i x_{ij}^{k} \leq Q_k, \quad k = 1, \dots, M; j = 0,1, \dots, n+m \qquad (1.19)$$

$$\sum_{i=n+1}^{n+m}\sum_{j=1}^{n+m} f_i x_{ij}^{k} \leq Q_k, \quad k = 1, \dots, M; j = 0,1, \dots, n+m \qquad (1.20)$$

$$\sum_{i=0}^{n+m} x_{ij}^{k} - \sum_{l=0}^{n+m} x_{jl}^{k} = 0, \quad j = 1, \dots, n+m; \ k = 1, \dots, M \qquad (1.21)$$

$$\sum_{i=0}^{n+m} x_{ij}^{k} - \sum_{l=0}^{n+m} x_{li}^{k} = 0, \quad i = 1, \dots, n+m; \ k = 1, \dots, M \qquad (1.22)$$

$$\sum_{i=0}^{n+m} x_{ij}^{k} \leq |S| - 1, \quad S \subseteq \{2,3, \dots, n+m\} \qquad (1.23)$$

$$\sum_{i=n+1}^{n+m}\sum_{j=1}^{n}\sum_{k=1}^{M} x_{ij}^{k} = 0 \qquad (1.24)$$

$$x_{ij}^{k} \in \{0,1\}, i = 0,1,\dots,n+m; k = 1,\dots,M \qquad (1.25)$$

L'équation (1.14) signifie que l'objectif du problème est de minimiser la somme des coûts de toutes les tournées. Les équations (1.15) et (1.16) imposent que chaque client soit servi une et une seul fois. Les équations (1.17) et (1.18) assure que le nombre de véhicules quittant le dépôt égal au nombre de véhicules rentrent au dépôt. Les équations (1.19) et (1.20) vérifient que la capacité d'un véhicule après avoir visité un client. Les équations (1.21) et (1.22) assurent la continuité de la route (chaque liaison doit être servie par un seul véhicule). L'équation (1.24) assure la contrainte de précédence, les clients de livraison sont servis avant les clients de collecte. L'équation (1.23) évite la génération des semi-tournées et l'équation (1.25) assure les contraintes de binarité sur les variables de décision x_{ijk}.

1.3.4 VRPPD: Vehicle Routing Problem with Pickup and Delivery

Dans le problème VRPPD, à chaque client i est associé deux quantités : d_i représente la demande de livraison au client i et p_i la demande de produits à ramasser chez le client i.

Pour chaque client i, O_i désigne le sommet qui est à l'origine de la demande de livraison, et D_i représente le sommet qui est la destination de la demande de ramassage.

Dans le VRPPD, à chaque emplacement de client, la livraison est effectuée avant le ramassage, par conséquent, la capacité d'un véhicule avant son arrivé chez un client donné est définie par la charge initiale moins toutes les demandes déjà livrées plus toutes les demandes déjà ramassées [Berghida2015]. Le VRPPD peut être formulé comme suit [Desaulniers 2001] :

1.3.4.1 Formulation Mathématique du VRPPD

La formulation exige deux types de variables :

$$x_{ijk} = \begin{cases} 1 \text{ si } (i,j) \text{ est parcourue par le véhicule } k \\ 0 \text{ sinon} \end{cases}$$

La variable L_{ik} qui donne la charge du véhicule k après que le service dans le nœud $i \in V_k$ est achevé

La formulation est la suivante :

$$\textbf{Minimizer } z = \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij}^{k} x_{ij}^{k} \qquad (1.26)$$

Avec les contraintes suivantes :

$$\sum_{k\in K}\sum_{j\in N} x_{ij}^{k} = 1\,,(i \in P) \qquad (1.27)$$

$$\sum_{j\in N} x_{ij}^{k} - \sum_{j\in N} x_{n+i,j}^{k} = 0\,,(i \in P\,,k \in K) \qquad (1.28)$$

$$\sum_{j\in N} x_{0j}^{k} = 1\,,(k \in K) \qquad (1.29)$$

$$\sum_{j\in N} x_{ji}^{k} - \sum_{j\in N} x_{ij}^{k} = 0,(i \in P \cup D, k \in K) \qquad (1.30)$$

$$\sum_{i\in N} x_{i,2n+1}^{k} = 1\,,(k \in K) \qquad (1.31)$$

$$Q_j^k \geq (Q_i^k + q_j)x_{ij}^k,(i \in N, j \in N, k \in K) \qquad (1.32)$$

$$B_i^k + d_i + t_{i,n+i} \leq B_{n+i}^k,(i \in P, k \in K) \qquad (1.33)$$

$$\max\{0, q_i\} \leq Q_i^k \leq \min\{Q_k, Q_k + q_i\},(i \in N, k \in K) \qquad (1.34)$$

$$x_{ij}^k \in \{0,1\},(i \in N, j \in N, k \in K) \qquad (1.35)$$

L'équation (1.25) présente la fonction objective qui minimise la distance totale. L'équation (1.26) assure que chaque requête est servie exactement une fois et que les nœuds de livraison et de collecte associés sont visités par le même véhicule. Les équations (1.27) (1.28) (1.29) (1.30) assurent que chaque véhicule commence sa tournée au dépôt initial et termine au dépôt destination. L'équation (1.31) assure que la contrainte de capacité de véhicule est toujours respectée. L'équation (1.32) assure que le véhicule visite le nœud de collecte avant le nœud de livraison. L'équation (1.33) impose la contrainte de capacité. L'équation (1.34) impose une contrainte de binarité sur la variable de décision x_{ij}^k.

1.3.5 OVRP : Open Vehicle Routing Problem

Dans le problème de tournée de véhicules ouvert (OVRP) un véhicule ne revient pas au dépôt après avoir servi le dernier client sur son itinéraire, l'OVRP est envisagé dans la pratique dans la livraison à domicile de colis et journaux où les contractuels qui ne sont pas des employés de la société de livraison utilisent leurs propres véhicules et ne reviennent pas au dépôt. La formulation mathématique est décrite comme suit [Sariklis 2000]

1.3.5.1 Formulation Mathématique du problème OVRP

Soit n le nombre de clients, Soit N = {1, 2, . . ., N} l'ensemble des clients, Soit N_0 = {0, 1, . ., N} l'ensemble de tous les clients et le dépôt, qui est identifié par 0. Soit K = {1, 2, . ., K} l'ensemble des véhicules. Soit d_i la demande du client i. Soit l_h la charge du véhicule $i \in N$. Soit $max\ d_i \leq max\ l_h$.

Soit les variables de décision :

$$y_{ih} = \begin{cases} 1 & si\ le\ point\ i\ (client\ ou\ depot) est\ visité\ par\ le\ véhicule\ h \\ 0 & sinon \end{cases} \quad (1.36)$$

$$x_{ijh} = \begin{cases} 1 & si\ le\ véhicule\ h\ voyage\ du\ sommet\ i\ au\ sommet\ j \\ 0 & sinon \end{cases} \quad (1.37)$$

Soit c_{ij} $le\ coût\ de\ transport\ entre\ les\ clients\ i\ et\ j$.

Fonction objective :

$$\textbf{Minimizer}\ z = \sum_{i=0}^{n}\sum_{j=1}^{n}\sum_{h=1}^{k} c_{ij}x_{ijh}\ , \quad (1.38)$$

Sous les contraintes suivantes :

$$\sum_{i=0}^{n} d_i y_{ih} \leq l_h \qquad \forall\ h, \quad (1.39)$$

$$\sum_{h=1}^{k} y_{ih} = 1 \quad i = 1,2, \ldots n, \quad (1.40)$$

$$\sum_{i=0}^{n} x_{ijh} = y_{jh} \; j = 1,2,\dots n \qquad \forall h, \qquad (1.41)$$

$$\sum_{j=1}^{n} x_{ijh} = y_{jh} \; i = 0,1,2,\dots n, \qquad \forall h, \qquad (1.42)$$

$$x_{ijh} = 0 \; ou \; 1 \quad i,j = 1,2,\dots n; \qquad \forall h, \qquad (1.43)$$

$$y_{ih} = 0 \; ou \; 1 \quad i,j = 1,2,\dots n; \qquad \forall h. \qquad (1.44)$$

L'équation (1.38) représente la fonction objective qui minimise la distance totale traversée. L'équation (1.38) est la contrainte qui porte sur la capacité des véhicules, L'équation (1.39) indique que chaque nœud de demande doit être servi, Les équations (1.40) et (1.41) assurent que chaque nœud de demande est servi par exactement le même véhicule, Les équations (1.42) et (1.43) assurent les contraintes de binarité sur les variable de décision x_{ijh} et y_{ih}

1.4 Conclusion

Ce chapitre présente une description du problème de VRP ainsi que certaines variantes de ce dernier et leurs formulations mathématiques y compris le problème étudié dans ce travail le CVRP. Dans le chapitre qui suit nous allons présenter comment résoudre grâce à une approche coopérative le problème de tournées de véhicules avec contrainte de capacité CVRP.

2. Chapitre 02 : Résolution du problème de tournées de véhicules avec contrainte de capacité (CVRP)

2.1 Introduction

Le VRP appartient au noyau de planification industrielle mais aussi à la grande famille des problèmes combinatoires NP-difficiles (NP-hard Problem) [HAJ 2010]. Ceci signifie qu'il n'y a pas de méthode exacte pouvant résoudre les problèmes qui incluent un grand nombre de clients avec une durée de temps de calcul raisonnable [Zhu, 1999]. Un problème d'optimisation combinatoire NP-difficile est difficile à résoudre de manière exacte pour les grandes instances. Les méthodes exactes nécessitent un grand temps de calcul qui croit exponentiellement avec la taille des instances du problème. C'est pour cela qu'on fait appel aux méthodes approchées (heuristiques et Métaheuristiques) [HAJ 2010].

Dans ce cas, il est nécessaire d'utiliser des méthodes approchées. Ainsi, les méthodes de résolution consacrées aux VRP et à ses variantes sont principalement classées en deux catégories : les méthodes exactes et les méthodes approchées. Ces dernières sont aussi divisées en deux sous-groupes : les heuristiques et les métaheuristiques [HAJ 2010].

Dans ce chapitre on va étudier la version de base de VRP (CVRP). Le CVRP (Capacitated Vehicle Routing Problem) consiste à affecter chaque client à une tournée effectuée par un seul véhicule de capacité finie. Ce véhicule commence et termine sa tournée au dépôt [Ralphs, 2003].

Dans ce qui suit, nous citons les principaux travaux qui traitent le problème du CVRP par les méthodes approchées. En 2010 un algorithme quantique d'optimisation par essaims particulaires est proposé dans [Zhengchu 2010] pour résoudre le problème de CVRP. De même, pour résoudre la même variante (CVRP), un algorithme de colonies de fourmis modifié est proposé dans [Chen 2012a]. Il consiste à minimiser la distance totale parcourue par chaque véhicule ainsi que le temps total de service sur les nœuds clients. Le CVRP étudié est traité en deux phases : phase d'affectation client/véhicule et phase de minimisation du temps d'exécution.

2.2 Technique de coopération proposée

Pour résoudre un problème d'optimisation, une méthode qui paraît bien adaptée est choisie parmi celles existantes (méthode exacte, heuristiques,...etc.).
Ensuite, des essais d'amélioration sont apportés en travaillant sur ses paramètres afin d'obtenir la méthode la plus efficace possible. Si c'est possible, sa qualité est évaluée en la comparant à d'autres méthodes proposées pour le problème étudié. Malheureusement, d'après les No Free

Lunch Theorems [Wolpert 1997], il n'existe pas de métaheuristique qui soit meilleure que toutes les autres métaheuristiques pour tous les problèmes. Dans la pratique, il existera toujours des instances pour lesquelles une métaheuristique est meilleure qu'une autre. Quelques soit la métaheuristique choisie, elle présente des avantages et des inconvénients.

L'idée de coopérer les algorithmes de recherche est classique en optimisation Combinatoire. Dans la co-évolution coopérative CC [Coo 2014], les collaborations entre l'ensemble d'individus sont nécessaires afin d'évaluer une solution complète. CC a été présenté comme une approche évolutive alternative pour optimiser les fonctions [Potter 1994]. Pour cela, ils considèrent que le nombre de variables d'une fonction est égal au nombre de populations et chaque variable est un composant de la solution qui est traitée séparément en utilisant un algorithme évolutionnaire. Pour résoudre un problème en utilisant CC, tout d'abord, nous devons décomposer le problème en sous-problèmes. Chaque sous-problème est attribué à une population. Toutes les populations évoluent au même temps. Elles n'échangent les informations que dans l'étape d'évaluation des solutions (calcul de fitness). Chaque individu dans une population représente une partie de la solution et une solution candidate potentielle pour chaque composant (sous-problème). Il ne peut pas être évalué séparément des autres parties complémentaires. La coopération dans le domaine des métaheuristiques pourrait également prendre la forme d'algorithmes hybrides qui sont connus pour être supérieurs aux algorithmes de recherche pures concernant la qualité des optima trouvés et le temps pris pour les trouver [Preux 1999] [Fleurent 1994]. L'hybridation entre algorithmes prend généralement l'une des formes suivantes :

- **Hybride séquentiel** : lorsque deux algorithmes sont appliqués l'un après l'autre.
- **Hybride parallèle synchrone** : où un algorithme est utilisé à la place d'un opérateur.
- **Asynchrone hybride parallèle** : où plusieurs algorithmes de recherche travaillent concurremment et échangent des informations.

Les hyper-heuristiques peuvent être définies comme des méthodes d'optimisation ayant la caractéristique d'utiliser une approche heuristique pour sélectionner les heuristiques afin de résoudre un problème d'optimisation.

L'approche Hyper-heuristiques propose de regrouper un ensemble d'heuristiques ou métaheuristiques et d'établir un mécanisme pour identifier et sélectionner les méthodes de recherche les plus efficaces au cours du processus d'optimisation [Berghida 2015].

Généralement, certaines études visent à produire des heuristiques constructives qui construiront une solution, étape par étape. Les heuristiques sont utilisées pour décider comment étendre une solution partielle. Ces méthodes ont tendance à être rapide. D'autres

études visent à produire des heuristiques d'amélioration ou de perturbation travaillant sur une solution candidate déterminée et essayant d'améliorer sa qualité. Ces méthodes sont plus lentes mais fournissent les meilleurs résultats finaux. Certaines études sont des hybridations entre les deux méthodes [Berghida 2015].

Nous proposons dans ce mémoire une autre forme de coopération entre les métaheuristiques où une métaheuristique maitresse contrôle plusieurs métaheuristiques esclaves. Les points forts d'une métaheuristique esclave compensent les points faibles d'une autre grâce à la notion de rang. Le rang est un moyen de mettre en valeur les points forts des métaheuristiques esclaves. Il est mis à jour continument par la métaheuristique maitre qui augmente le rang des heuristiques ayant amélioré la solution et diminue le rang de celles qui ne l'auront pas amélioré. Cette mise à jour tend à récompenser les heuristiques esclaves qui auront réussi à améliorer la solution courante. Le rang aide l'heuristique maitresse à choisir la meilleure heuristique esclave à chaque point de décision en choisissant l'heuristique esclave ayant le plus grand rang. L'algorithme 2.1 présente le processus de coopération, où :

- Δ représente la différence de bénéfice entre la solution trouvée par l'une des métaheuristiques esclaves et la solution courante.
- α représente la différence entre la qualité de la solution trouvée et celle de la solution initiale. Si la qualité de la solution est améliorée, α sera positif, sinon α sera négatif.

La concurrence entre les métaheuristiques esclaves favorise l'intensification de la solution dans l'espace de recherche.

```
1   Debut
2       Tant que condition d'arrêt non vérifiée faire
3           Choisir la métaheuristique k, avec le plus haut rang et k E Liste Taboue.
4           Si Δ > 0 alors
5               r_k = r_k + α et vider la Liste Taboue.
6               Sinon
7                   r_k = r_k + α et inclure k dans la Liste Taboue.
8               Fin
9           Fin
10      Fin
11  Fin
```

Algorithme 2.1 : Algorithme de coopération

La coopération entre les esclaves est introduite par le mécanisme suivant : lorsqu'une métaheuristique termine sa recherche, elle communique sa meilleure solution à celle qui la remplace, permettant ainsi à cette dernière de générer un bon voisinage. La principale différence entre l'approche coopérative proposée et CC est le mécanisme d'activation de métaheuristiques pour chercher une solution. Dans notre cas, nous avons utilisé la notion de rang pour activer une métaheuristique, tandis que dans CC, l'exécution des métaheuristiques

est parallèle. Dans notre cas, il n'y a pas de décomposition de problème, chaque métaheuristique esclave manipule une solution complète qui peut être utilisée par d'autres métaheuristiques. Toutefois, CC traite des sous-problèmes où chaque composant gère une partie de la solution. La différence principale entre les hyper-heuristiques et l'approche coopérative proposée que nous manipulions des solutions complètes, tandis que les hyper-heuristiques manipulent des solutions partielles.

2.3 Encodage de la solution

2.3.1 Définition de la structure de données

Le codage est l'une des étapes les plus importantes dans la résolution d'un problème, car la manière de représenter une solution à une grande influence sur la qualité des résultats, c'est pourquoi il faut choisir le codage le plus adapté au problème résolu. Dans notre travail nous proposons de coder la solution sous forme d'un vecteur de taille K tel que K représente le nombre total des véhicules. Chaque itinéraire est représenté par un véhicule. Chaque élément du vecteur est une liste de clients visités par le véhicule.

Exemple : supposant qu'on procède 25 clients (codés de 1 à 25) et 5 véhicules codés de (V1 à V5). Une solution réalisable peut être présentée comme suit :

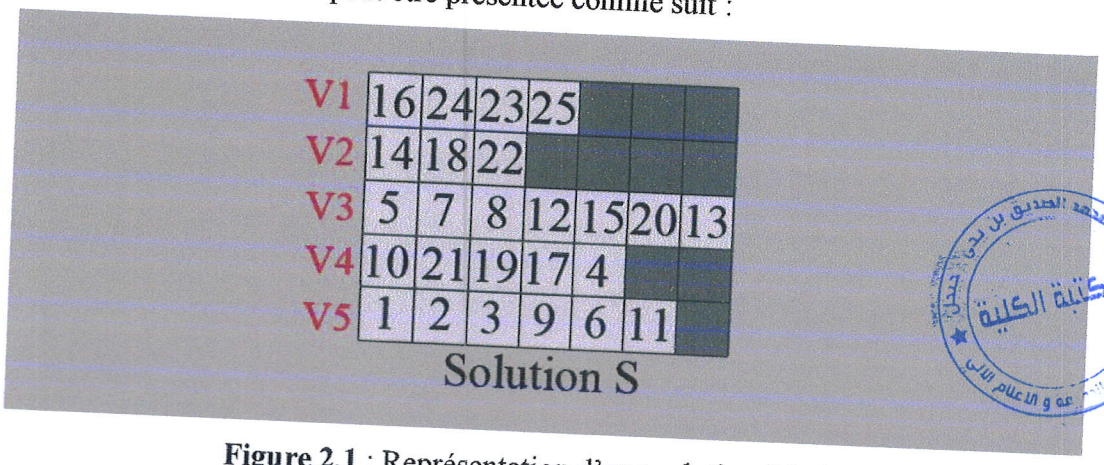


Figure 2.1 : Représentation d'une solution faisable

Les clients 16, 24, 23, 25 sont visités par le véhicule 1 dans cet ordre. Le véhicule 2 a visité les clients 14 et 18 puis le client 22, le véhicule 3 a visité les clients 5, 7, 8, 12, 15, 20, 13, et le véhicule 4 a visité les clients 10, 21, 19, 17, 4, respectivement, et pour terminer le véhicule 5 a visité les clients 1, 2, 3, 9, 6, 11.Chaque tournée se commence et se termine dans le dépôt. Pour chaque véhicule, la contrainte de capacité doit être respectée.

2.3.2 Génération d'une solution

Au démarrage de chaque processus de recherche, la génération des solutions initiales est une phase importante. Les choix standards pour générer une solution initiale sont : soit une solution initiale aléatoire ou une solution retournée par une heuristique. Dans un problème de

tournées de véhicules, la façon la plus simple est d'affecter aléatoirement chaque client à un véhicule tout en respectant les contraintes du problème. Dans le problème CVRP, pour générer une solution faisable aléatoirement nous commençons par un véhicule et nous affectons des clients de maniéré aléatoire à ce véhicule. Lorsque le véhicule est rempli ou bien saturé, nous ajoutons un autre véhicule et ainsi de suite jusqu'à ce que tous les clients soient servis. Le nombre de véhicule ne doit pas être dépassé.

La procédure de génération est décrite par l'algorithme qui suit :

```
1   Debut
2       Trier les clients par ordre décroissant selon leurs demandes d_i
3       Répéter
4           Pour i allons de 1 à NBR_Vehicule Faire
5               Choisir un véhicule V_i;
6               Tant que chargeV_i <= capacity_V_i Faire
7                   Remplir V_i avec les demandes de livraisons d_i;
8                   chargeV_i := chargeV_i + d_i;
9               Fin
10          Fin
11      Jusquà tous les clients sont servis
12  Fin
```

Algorithme 2.2 : Création d'une solution

2.3.3 Rectification d'une solution

La génération des solutions initiales aléatoirement garantit toujours la faisabilité des solutions générées. Par contre dans le processus de modification des solutions dans l'algorithme BBO, la faisabilité des nouvelles solutions n'est pas toujours garantie en termes de capacité et de satisfaction des demandes des clients. Pour résoudre ce problème une méthode de rectification s'avère indispensable. La procédure de rectification est décrite dans l'algorithme 2.8.

2.4 Métaheuristiques utilisées dans la coopération

Dans cette approche nous avons utilisé trois métaheuristiques.

2.4.1 L'optimisation basée sur la biogéographie BBO

La première métaheuristique utilisée est l'optimisation basée sur la biogéographie BBO.

2.4.1.1. BBO : Biogeography Based Optimization (Optimisation Basée sur la Biogéographie)

BBO est un nouvel algorithme inspiré de la biogéographie pour l'optimisation globale. La biogéographie étudie la répartition géographique d'organismes biologiques. Il est dû aux

travaux d'Alfred Wallace [Wallace 1876a], [Wallace 1876b] et Charles Darwin [Darwin 1995] dans le 19éme siècle. Ces travaux ont eu un aspect descriptif et un aspect historique. En 1960, RobertMacArthur et Edward Wilson ont développé un modèle mathématique pour la biogéographie [MacArthur 1967].

Les modèles mathématiques de la biogéographie décrivent comment les espèces migrent d'une île à l'autre, comment de nouvelles espèces apparaissent et comment les espèces s'éteignent [Simon 2008].

Une île est un habitat, un habitat est un espace de vie isolé des autres espaces. Les habitats qui sont favorables à la résidence des espèces biologiques ont un HSI (Habitat Suitability Index) élevé. Les paramètres influençant le HSI peuvent être : la pluie, la diversité végétale, la diversité des terrains ...etc.

Les variables décrivant l'habitabilité sont appelées SIV (Suitability IndexVariables). Les habitats sont caractérisés par ce qui suit :

- Un habitat avec un HSI élevé tend à avoir un nombre élevé d'espèces, alors que celui avec un HSI bas tend à avoir moins d'espèces.
- Les habitats avec un HSI élevé sont caractérisés par un taux élevé d'émigration Et un taux bas d'immigration parce qu'ils sont saturés.
- Les habitats avec un HSI bas ont un taux élevé d'immigration et un taux bas d'émigration. L'immigration doit entraîner la modification du HSI de l'habitat. Les espèces qui vivent dans un habitat qui reste trop longtemps sans amélioration ont tendance à disparaître.

Dan Simon a introduit en 2008 une métaheuristique basée sur la biogéographie. Il utilise l'analogie suivante :

- Une solution est analogue à un habitat.
- La qualité (fitness) d'une solution est analogue au HSI.
- Les variables définissant la solution sont analogues au SIVs.
- Une bonne solution est analogue à un habitat avec un HSI élevé, un nombre élevé d'espèces, un haut taux d'émigration et un taux bas d'immigration.

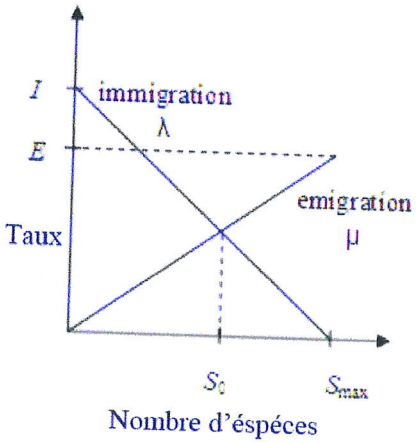


Figure 2.2 : Evolution des taux d'émigration et immigration en fonction du nombre d'espèces [Berghida 2015]

- Une mauvaise solution est analogue à un habitat avec un HSI bas, un nombre bas d'espèces, un taux bas d'émigration et un taux élevé d'immigration.
- Une bonne solution tend à partager ses caractéristiques avec une mauvaise Solution pour l'améliorer (migration des SIVs). Ceci est analogue à la migration des espèces entre les habitats. Partager les caractéristiques n'entraine pas un changement dans les caractéristiques de la bonne solution, puisque la migration se fait en utilisant seulement un échantillon d'espèce.
- Les mauvaises solutions acceptent les caractéristiques des bonnes solutions pour améliorer leur qualité. Ceci est analogue au mauvais habitat qui accepte immigration des espèces depuis d'autres habitats.

La figure 2.2 illustre un modèle d'abondance des espèces dans un habitat. Elle présente des courbes d'immigration et d'émigration comme une ligne droite (modèle simple), ce qui nous donne une description générale du processus d'immigration et d'émigration.

Comme décrit dans l'algorithme de migration (Algorithme 2.3), supposons que nous ayons une population de solutions candidates à un problème, représentées par des vecteurs (habitatsH_i , $i = 1 \dots n$). Chaque élément du vecteur est considéré comme une valeur SIV. Dans le processus de migration, les caractéristiques des bonnes solutions remplacent les mauvaises en utilisant le taux d'immigration λ(2.1) et le taux d'émigration μ (2.2) selon la probabilité de changement P_{mod}.

$$\lambda_k = I\left(1 - \frac{K}{N}\right) \qquad (2.1)$$

$$\mu_k = E \times \frac{K}{N} \qquad (2.2)$$

Où :

λ_k : est le taux d'immigration dans un habitat de k espèces.

μ_k : est le taux d'émigration dans un habitat de k espèces.

E : est le taux maximal d'émigration.

I : est le taux maximal d'immigration.

N : est le nombre maximum d'espèces.

K : est le nombre d'espèces.

Dans le processus de migration (algorithme 2.3), quand une solution est sélectionnée pour être changée, nous utilisons le taux d'immigration λ pour décider si un SIV sera changé. Dans ce cas, nous utilisons le taux d'émigration μ pour décider quelle bonne solution migrera son SIV.

```
1   Debut
2       Sélectionner un habitat H_i avec une probabilité αλ_i
3       Si H_i est sélectionné alors
4           Pour j=1 to n faire
5               Sélectionner un habitat H_j avec une probabilité αμ_j
6               Si H_j est selectionné alors
7                   Remplacer un SIV dans H_i avec un SIV de H_j
8               Fin
9           Fin
10      Fin
11  Fin
```

Algorithme 2.3 : Algorithme de migration

Dans le processus de mutation (Algorithme 2.4), la mutation est exécutée pour toute ou une partie de la population. D'une manière similaire à la mutation dans les algorithmes génétiques. Des changements dans un habitat peuvent survenir. Ces modifications changeront le HSI de l'habitat. Ceci est modélisé dans BBO par la mutation avec une certaine probabilité. Le taux de changement de l'habitat est donné dans la formule (2.3).

$$m(s) = \alpha \times \frac{1 - P_s}{P_{max}} \quad (2.3)$$

Où :

$m(s)$: est le taux de mutation d'un habitat avec s espèces.

α: est un paramètre défini par l'utilisateur

P_s: est la probabilité d'avoir s espèces dans l'habitat (formule 2.4).

P_{max} : est la probabilité d'avoir le nombre maximale d'espèces.

Et :

$$P_k = \{ \quad \frac{\lambda_0\lambda_1 \dots \lambda_{k-1}}{\mu_0\mu_1 \dots \mu_k \left(1 + \sum_{l=1}^{n} \frac{\lambda_0\lambda_1 \dots \lambda_{l-1}}{\mu_0\mu_1 \dots \mu_l}\right)} \quad 1 \le K \le n \quad (2.4)$$

$$\frac{1}{1 + \sum_{l=1}^{n} \frac{\lambda_0\lambda_1 \dots \lambda_{l-1}}{\mu_0\mu_1 \dots \mu_l}} k = 0$$

Remarque : Les habitats à muter sont ceux qui ont une valeur basse de HSI. Cette mutation introduit la diversité et encourage les habitats moyens à s'améliorer.

```
Debut
    Pour j=1 à m faire
        Utiliser λ_i et μ_i pour calculer P_i
        Sélectionner un SIV j de H_i (H_i(j)) avec une probabilité αP_i
            Si H_i(j) est sélectionné alors
                Remplacer H_i(j) par un SIV aléatoire
            Fin
    Fin
Fin
```

Algorithme 2.4 : Algorithme de mutation

Dans les stratégies évolutionnaires, la recombinaison est utilisée pour créer de nouvelles solutions, alors que la migration dans BBO est utilisée pour modifier les solutions existantes. La recombinaison globale dans les stratégies évolutionnaires est un processus reproductif, alors que la migration dans BBO est un processus adaptatif ; elle est utilisée pour modifier les habitats existants.

Comme dans la plupart des algorithmes d'optimisation à base de population, l'élitisme est typiquement incorporé dans le but de garder la meilleure solution dans la population. Cela empêche que les meilleures solutions soient altérées par l'immigration.

Pour éviter de perdre les bonnes solutions après les processus de migration et de mutation. Nous copions une ou plusieurs des meilleures solutions dans la nouvelle génération. Dans notre cas, nous choisissons de copier les deux meilleures solutions.

Nous allons dans ce qui suit décrire la première approche développée en implémentant une version améliorée de BBO appelée BBO-A.

Debut

1. Initialiser un ensemble de solutions au problème.
2. Calculer le HSI de chaque solution.
3. Calculer s, λ, μ de chaque solution.
4. Modifier les habitats (migration basée sur λ, μ).
5. Mutation basée sur la probabilité.
6. Typiquement, on applique l'élitisme.
7. Répéter à partir de l'étape 2 pour l'itération suivante.

Fin

Algorithme 2.5 : Algorithme BBO

2.4.1.2 BBO-A : Algorithme d'Optimisation Amélioré Basée sur la Biogéographie pour le CVRP

Nous proposons de résoudre CVRP en utilisant un algorithme amélioré de BBO. Cette amélioration est effectuée dans le processus de mutation, à chaque itération la tournée cible est modifiée de manière ce que le véhicule commence du dépôt et se déplace vers le plus proche client et ainsi de suite.

L'approche proposée BBO-A est présentée dans l'algorithme qui suit :

Debut

1. Réglage des paramètres : nombre d'itérations, la taille de la population, I, E.
2. Créer un ensemble initial d'habitat satisfaisant les contraintes de problème.
3. Calculer le HSI de chaque habitat.
4. Calculer s,λ,μ de chaque habitat.
5. Modifier les habitats (migration basée sur λ,μ) satisfaisant les contraintes de problème et en considérant la tournée comme SIV.
6. Mutation basée sur la probabilité en permutant entre les clients de la même tournée de manière Que le véhicule se déplace toujours vers le plus proche client.
7. Appliquer l'élitisme.
8. Répéter à partir de l'étape 3 pour l'itération suivante.

Fin

Algorithme 2.6 : Algorithme BBO-A

2.4.1.2.1 Description de l'algorithme

Réglage des paramètres : Dans la première étape, nous proposons à l'utilisateur une interface de réglage de paramètres initiale de BBO-A qui sont :

- I : le taux d'immigration
- E : le taux d'émigration
- Taille de population
- Nombre d'itérations

Création d'une population initiale d'habitats : Dans la deuxième étape, nous générons un ensemble initial d'habitats (algorithme 2.7). Chaque habitat code une solution possible. Chaque solution (habitat) est générée aléatoirement (solution faisable).

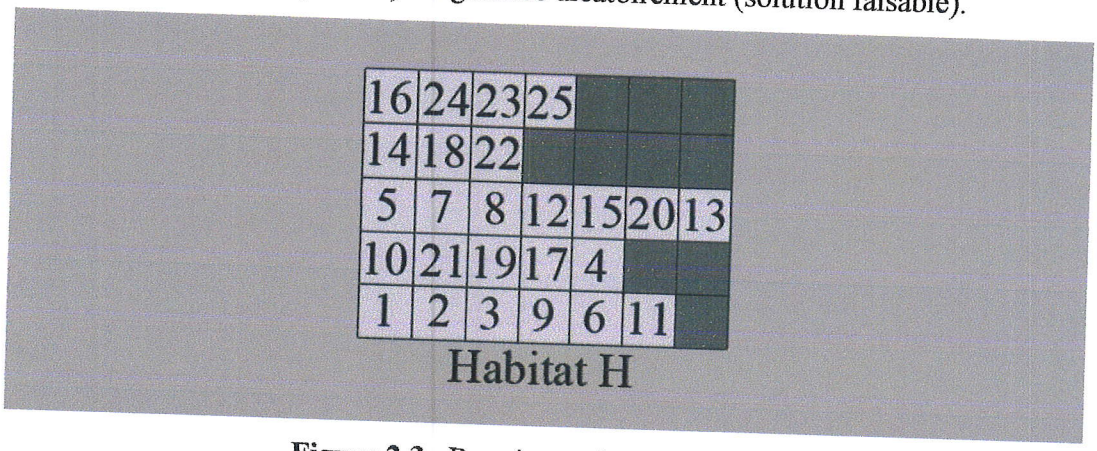


Figure 2.3 : Représentation d'un habitat

```
Debut
    Tant que m <= nombre de population Faire
        m := m+1
        Trier les clients par ordre décroissant selon leurs demandes d_i
        Répéter
            Pour i allons de 1 à NBR_Vehicule Faire
                Choisir un véhicule V_i;
                Tant que chargeV_i <= capacity_V_i Faire
                    Remplir V_i avec les demandes de livraisons d_i;
                    chargeV_i := chargeV_i + d_i;
                Fin
            Fin
        Jusquà tous les clients sont servis
    Fin
Fin
```

Algorithme 2.7 : Création d'une population initiale

Calcul du HSI Après la création de l'ensemble des habitats, chaque habitat est évalué et se voit assigné une valeur de fitness (HSI), selon la fonction de fitness suivante :

$$HSI = \sum_{r=1}^{k} (distance\ parcorue\ par\ le\ véhicule\ r) \qquad 2.5$$

Calcul de s, Λ et μ :

Calcul de *s* (nombre d'espèces) : Les bonnes solutions sont caractérisées par un nombre élevé d'espèces. Pour associer un nombre d'espèces pour chaque solution, nous trions les solutions par ordre décroissant de la valeur du fitness. Nous associons le rang de la solution au nombre d'espèces. Alors la meilleure solution aura le nombre d'espèces le plus élevé et la plus mauvaise le plus bas.

Calcul de λ (taux d'immigration) et μ (taux d'émigration) : Les variables λ et μ caractérisant l'habitabilité sont calculées comme suit :

$$\lambda_k = I\left(1 - \frac{K}{N}\right) \qquad (2.6)$$

$$\mu_k = E \times \frac{K}{N} \qquad (2.7)$$

Tel que I et E sont des constantes qui représentent le taux maximal d'immigration et d'émigration respectivement. n est le nombre total d'habitats.

Migration basée sur λ et μ : L'opérateur de migration λ décrit dans l'algorithme 2.3 est utilisé pour partager les informations des bonnes solutions et l'opérateur d'immigration μ est utilisé pour partager les informations vers les mauvaises solutions.

Exemple : Soient Sol_1 et Sol_2 deux solutions générées dans la même itération. Supposons que Sol_1 est une mauvaise solution sélectionnée pour la migration et Sol_2 une bonne solution sélectionnée pour améliorer Sol_1. Les deux solutions sont données dans les figures qui suivent.

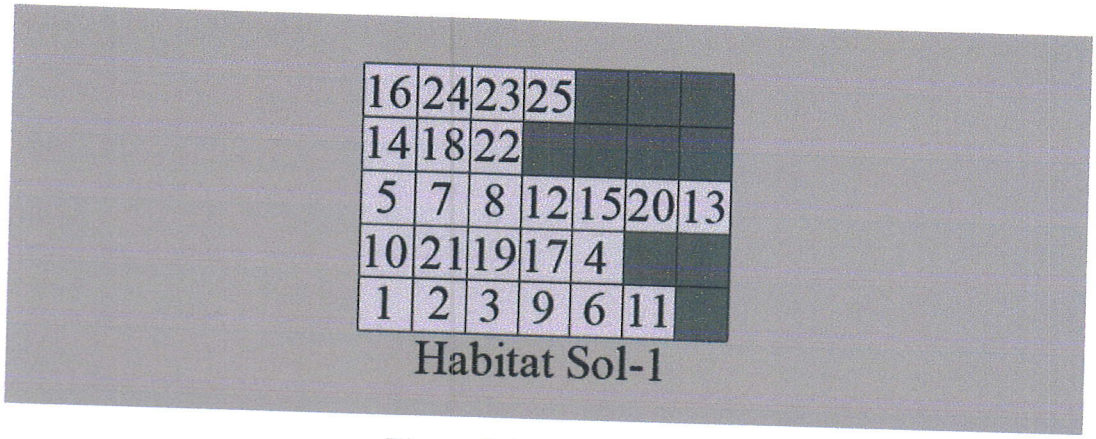


Figure 2.4 : Habitat Sol1

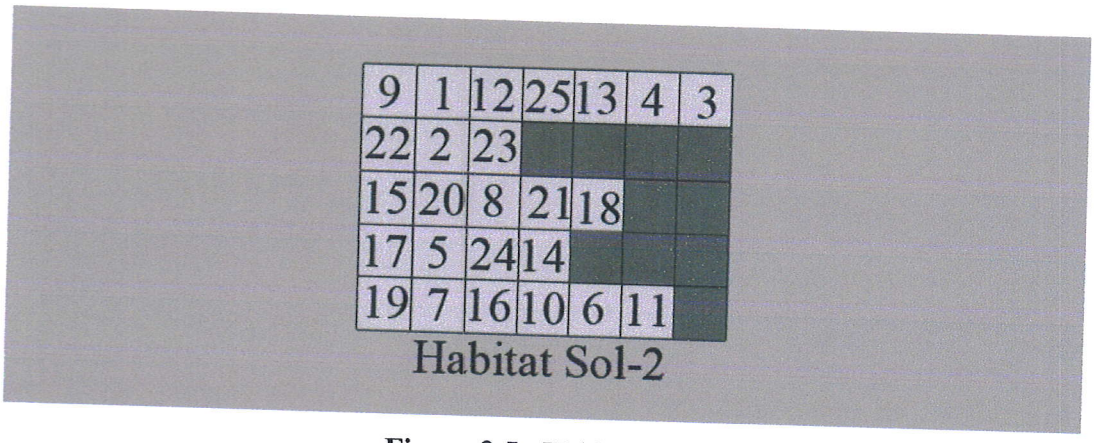


Figure 2.5 : Habitat Sol2

Nous décrivons dans ce qui suit le processus de migration : nous sélectionnons aléatoirement un SIV de Sol_1 (supposons que c'est V5). Nous sélectionnons aléatoirement un SIV de Sol_2 (supposons que c'est V1). On supprime V5 de Sol_1 et on le remplace par V1 de Sol_2 à la fin de la solution Sol_1. La nouvelle solution est représentée dans la figure qui suit :

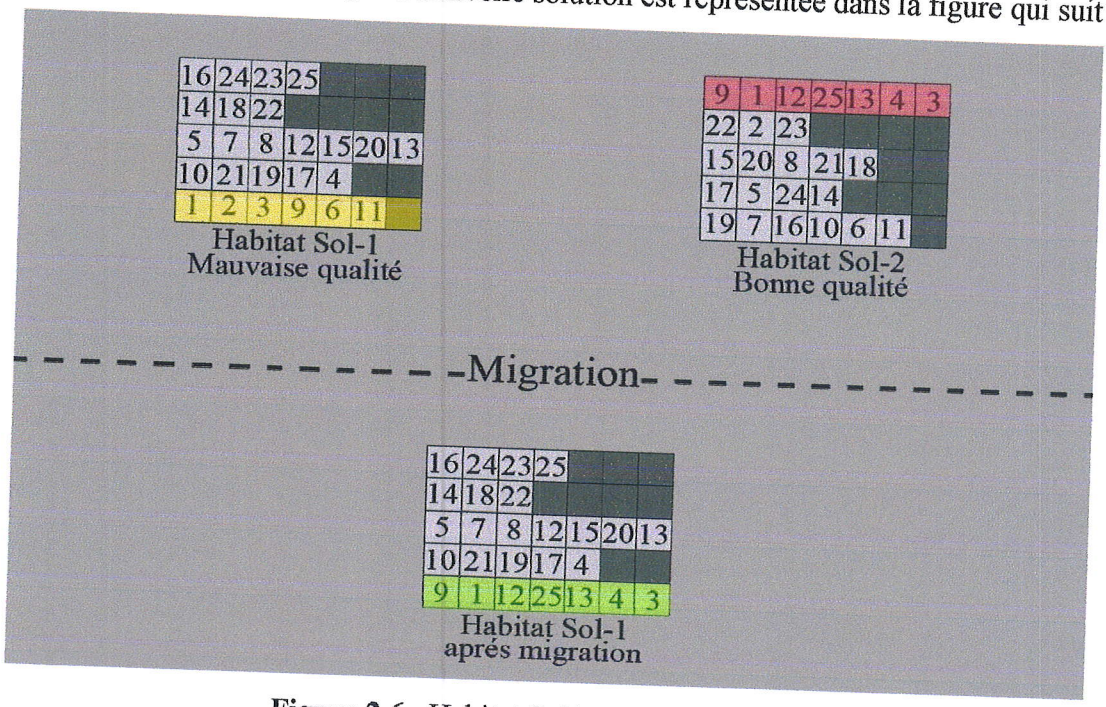


Figure 2.6 : Habitat Sol1 après la migration

On remarque que les clients 12, 25, 13 et 4 (doublons) sont visités par d'autres véhicules plus le véhicule V5 et que les clients 2, 6 et 11 qui appartient au SIV remplacé et ne figurent pas dans le SIV migré (clients qui manques). Une méthode de rectification s'avère indispensable. Le SIV ayant migré est maintenu donc les clients 12, 25, 13 et 4 qui existent déjà serons supprimés des autres véhicules. Après avoir éliminés les doublons, on ajoute les clients 2, 6 et 11 d'une façon aléatoire en respectant la contrainte de capacité.

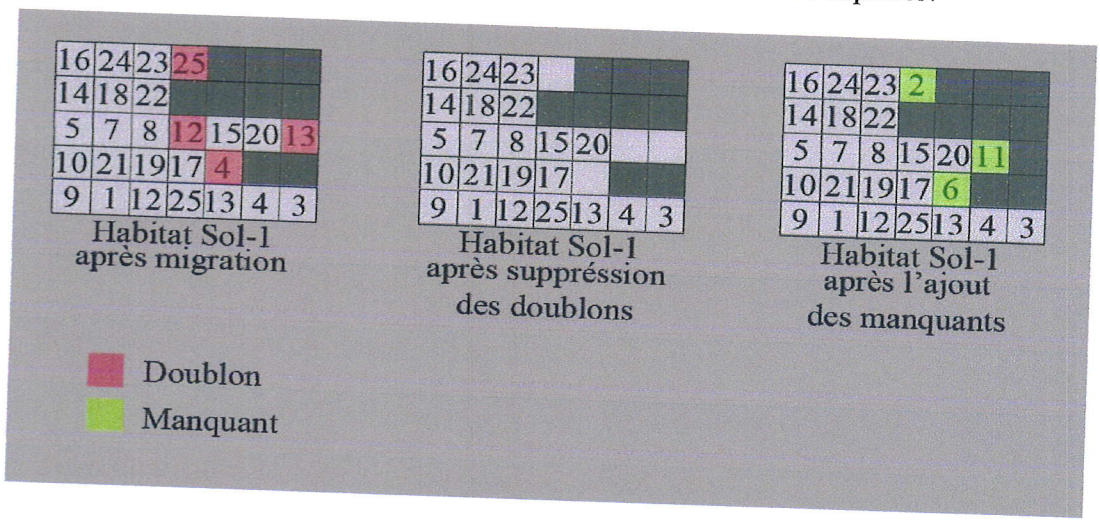


Figure 2.7 : Processus de rectification de solution

```
Debut
    Entrée : solution S,V_supprimé ;
    i:=0 ;
    Sauvegarder les clients manquants dans une liste M
    Sauvegarder les clients doublant dans une liste D
    Pour chaque SIV ≠ S_(V_migré ) Faire
        Supprimer C ϵ D
        Tant que i ≤ nbrVehicules Faire
            Pour chaque client C_j ϵ M Faire
                Si S_(V_i) + demande(C_j) ≤ capacity Faire
                    Ajouté C_j dans S_(V_i)
                Fin
            Fin
        Fin
    Fin
Retourner solution S ;
Fin
```

Algorithme 2.8 : Rectification d'une solution

Mutation : L'opérateur de mutation a tendance à augmenter la diversité de la population en prospectant d'autres régions de l'espace de recherche. Le processus de mutation consiste à changer l'ordre des clients dans chaque SIV de manière à ce que le véhicule parcoure les clients du dépôt vers le plus proche client et ainsi de suite.

Exemple : Supposons que le SIV de l'habitat Sol_1 représenté dans la figure qui suit est une tournée candidate pour la mutation.

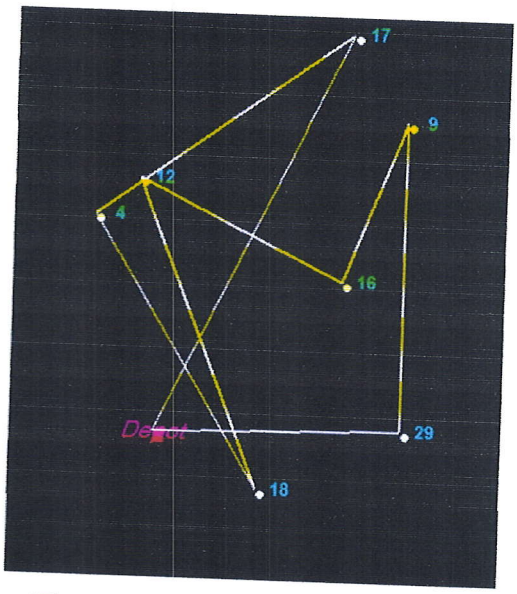


Figure 2.8 : SIV avant la mutation

Le processus de mutation permute dans un SIV les clients de manière à choisir le plus proche client toujours. La figure qui suit représente le résultat après la mutation.

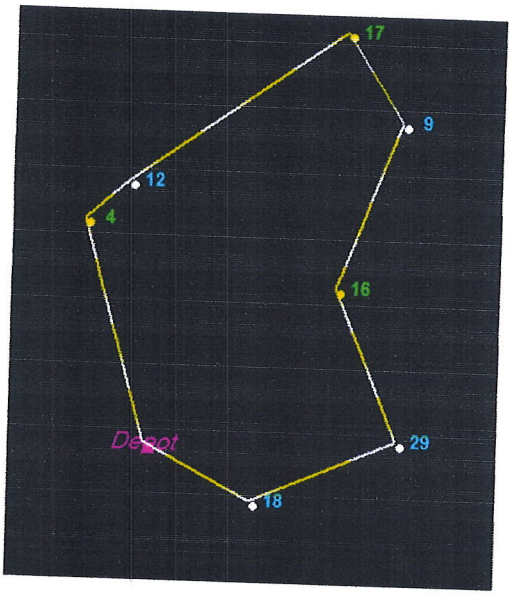


Figure 2.9 : SIV après la mutation

2.4.2 Local Search (La recherche locale simple (la descente))

2.4.2.1 LS : Local Search (La recherche locale)

La recherche locale simple ou la descente est un algorithme d'amélioration très ancien. Son principe consiste à explorer le voisinage de la solution courante afin d'améliorer sa qualité progressivement, comme le montre la figure 2.10 qui représente un schéma d'évolution d'une recherche locale simple. A chaque itération du processus d'amélioration, l'algorithme modifie un ensemble de composantes de la solution courante pour permettre le déplacement vers une solution voisine de meilleure qualité [Amira 2013].

Le processus est répété itérativement jusqu'à la satisfaction du critère d'arrêt. Il est à noter qu'il existe trois types de la descente : la descente déterministe, la descente stochastique et la descente vers le premier meilleur voisin [Amira 2013].

L'algorithme 2.9 résume les étapes de l'algorithme général de la descente, l'algorithme entame la recherche par la construction d'une solution initiale s et l'évaluation de sa qualité f(s). Ensuite il commence l'ensemble des étapes du processus d'amélioration suivantes [Amira 2013] :

- Modifier s pour obtenir une nouvelle solution s' de meilleure qualité que s et qui appartient à son voisinage. Le choix de la fonction de voisinage est important. Il dépend de l'objectif à atteindre. En fait, si l'objectif est de minimiser le coût on doit suivre la direction de la vallée. Sinon (dans le cas de maximisation), on doit suivre la direction du sommet.
- Evaluer la qualité f (s') de la nouvelle solution s'.
- Remplacer s par s', si s' est de meilleure qualité.

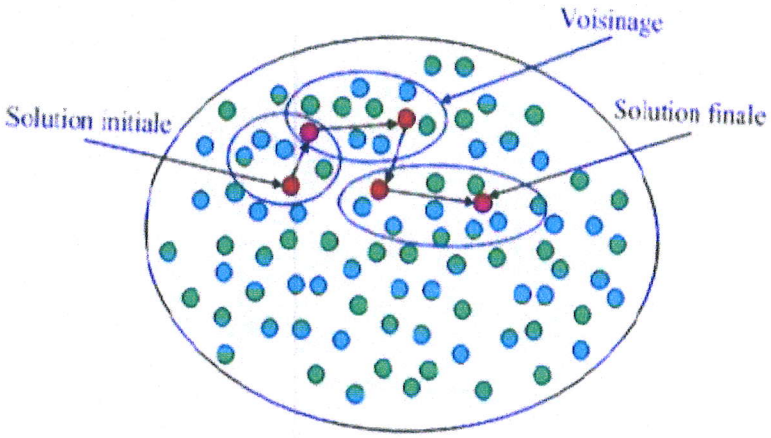


Figure 2.10 : Un schéma d'évolution d'une recherche locale simple [Naimi ,2008]

Le processus d'amélioration sera répété jusqu'à arriver au cas où toutes les voisines candidates sont de piètre qualité que la solution courante. Autrement dit, la recherche s'arrête lorsqu'un optimum local est atteint [Amira 2013].

L'avantage de la recherche locale simple revient à sa simplicité et à sa rapidité. Cependant, son problème réside dans le fait d'être bloquée par le premier optimum local rencontré. Ce dernier n'est pas forcement l'optimum global. Il peut être très loin de l'optimum global [Amira 2013].

Les bases de la recherche locale peuvent être définies ainsi. Soient [Amira 2013] :

– f () la fonction qu'on l'on souhaite maximiser,

– SLasolution courante,

– S^*la meilleure solution connue,

– f^*la valeur de la meilleure solution connue,

– N(S) le voisinage de S.

La procédure de recherche locale est la suivante :

```
1   Debut
2       Initialisation : construire une solution de départ S_0 ;
3       S*=S_0 ;
4       S =S_0 ;
5       f*= f (S_0) ;
6       Tant que l'optimum local n'a pas été atteint faire
7           S = argmax_{s'∈N(S)}(f(S')) ;
8           Si f(S) > f* alors
9               f*= f(S) ;
10              S*=S;
11          Fin
12      Fin
13  Fin
```

Algorithme 2.9 : Algorithme d'une recherche locale simple de plus grande pente

2.4.2.2 MLS : Multi-Start Local Search (La recherche locale à base de population) pour CVRP

Nous proposons de résoudre le CVRP en utilisant l'algorithme de recherche locale multi-Start qui est similaire à l'algorithme de recherche locale décrit dans la section 2.4.2.1 avec une différence qui consiste à commencer par une population de solutions plutôt qu'une seul solution. Nous appliquons ensuite la recherche locale sur le meilleur individu de cette population.

L'approche proposée MLS est présentée dans l'algorithme qui suit :

```
Debut
1   Réglage des paramètres : la taille de la population.
2   Créer un ensemble initial de solutions satisfaisant les contraintes de problème.
3   Evaluation des solutions et choix de la meilleure solution de la population.
4   Application de la recherche local sur la meilleure solution sélectionné (Algorithme2.9)
Fin
```

Algorithme 2.10 : Algorithme MLS

2.4.2.2.1 Description de l'algorithme

Réglage des paramètres : Dans cette étape l'utilisateur a la main de définir la taille de la population initiale avec laquelle il commencera notre algorithme.

Création d'une population initiale de solutions : Dans la deuxième étape, nous générons un ensemble initial de solutions (algorithme 2.11). Chaque solution est générée aléatoirement (solution faisable).

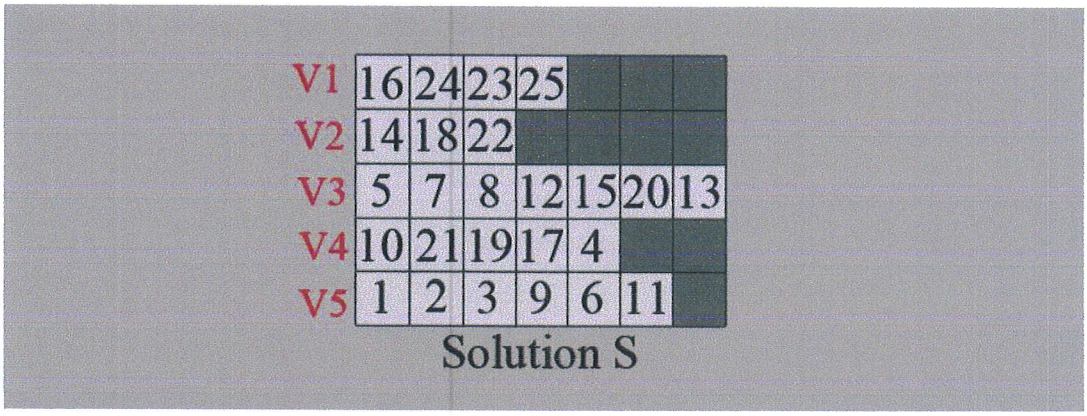


Figure 2.11 : Représentation d'une solution faisable

```
1   Debut
2       Tant que m <= nombre de population Faire
3           m := m+1
4           Trier les clients par ordre décroissant selon leurs demandes d_i
5           Répéter
6               Pour i allons de 1 à NBR_Vehicule Faire
7                   Choisir un véhicule V_i;
8                   Tant que chargeV_i <= capacity_V_i Faire
9                       Remplir V_i avec les demandes de livraisons d_i;
10                      chargeV_i := chargeV_i + d_i;
11                  Fin
12              Fin
13          Jusquà tous les clients sont servis
14      Fin
15  Fin
```

Algorithme 2.11 : Création d'une population initiale

Evaluation des solutions et choix de la meilleure solution de la population : Après avoir généré l'ensemble initial des solutions. Nous affectons à chaque solution une fitness qui va être considérer comme critère d'évaluation des solutions. Ce dernier est calculé par la formule qui suit :

$$Fitness = \sum_{r=1}^{k} (distance\ parcorue\ par\ le\ véhicule\ r) \qquad 2.8$$

La solution qui aura la meilleure valeur de fitness va être sélectionnée pour la prochaine étape.

Application de la recherche locale : Après avoir sélectionnée la meilleure solution de la population nous appliquons l'algorithme de recherche local décrit dans la section 2.4.2.1.

2.4.2.2.2 Méthode de voisinage utilisé dans l'algorithme MLS

Dans l'algorithme MLS, nous avons utilisé la méthode de voisinage 2-OPT.

2.4.2.2.2.1 Méthode de voisinage 2-OPT

L'algorithme de recherche locale 2-OPT a été mis en œuvre par Georges A. Croes en 1958 [Croes 1958] afin de résoudre le problème de voyageur de commerce ainsi que de nombreux problèmes connexes. Ceux-ci comprennent le problème de tournées de véhicules (VRP). Dans notre cas nous proposons d'appliquer cet algorithme sur le problème CVRP.

L'algorithme commence initialement par une solution et l'idée principale derrière est de prendre une route qui traverse sur elle-même et la réorganiser afin de minimiser la distance du chemin parcourue.

L'approche 2-OPT est représentée dans l'algorithme qui suit :

```
1   Debut
2       Entré : Véhicule V
3       Amélioration := vrai ;
4       Tant que Amélioration = vrai faire
5           Amélioration := faux ;
6           Pour tout client C_i de V faire
7               Pour tout client C_j de V avec j ≠i+1 ∧ j ≠i-1 ∧ j ≠i faire
8                   Si distance(C_i, C_{i+1}) + distance(C_j, C_{j+1}) > distance(C_i, C_j) + distance(C_{i+1}, C_{j+1}) alors
9                       permutation entre les clients C_{i+1} et C_j
10                      Amélioration := vrai ;
11                  Fin
12              Fin
13          Fin
14      Fin
15      Retourner V
16
17  Fin
```

Algorithme 2.12 : Algorithme général 2-OPT

Les figures qui suivent illustrent le fonctionnement de l'algorithme 2-OPT :

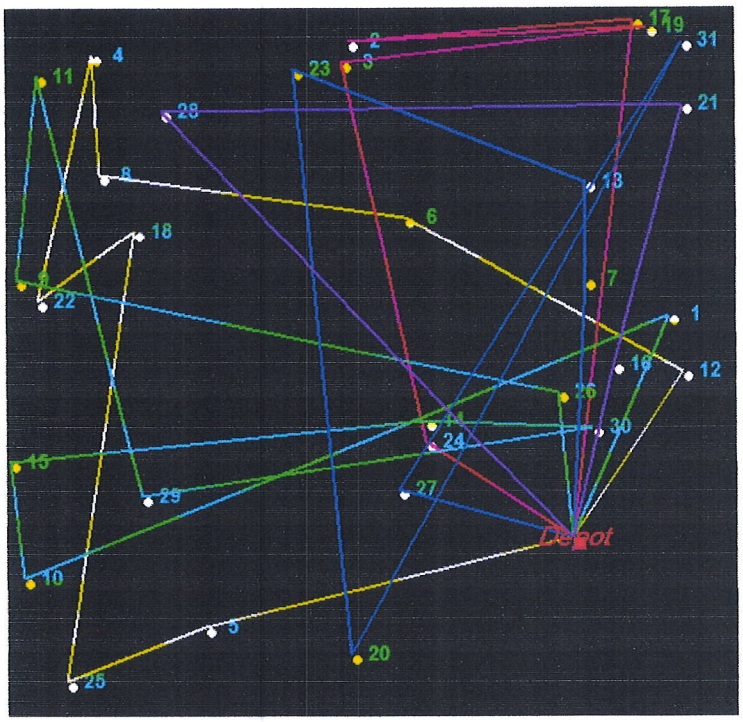


Figure 2.12 : Avant l'exécution de l'algorithme 2-OPT

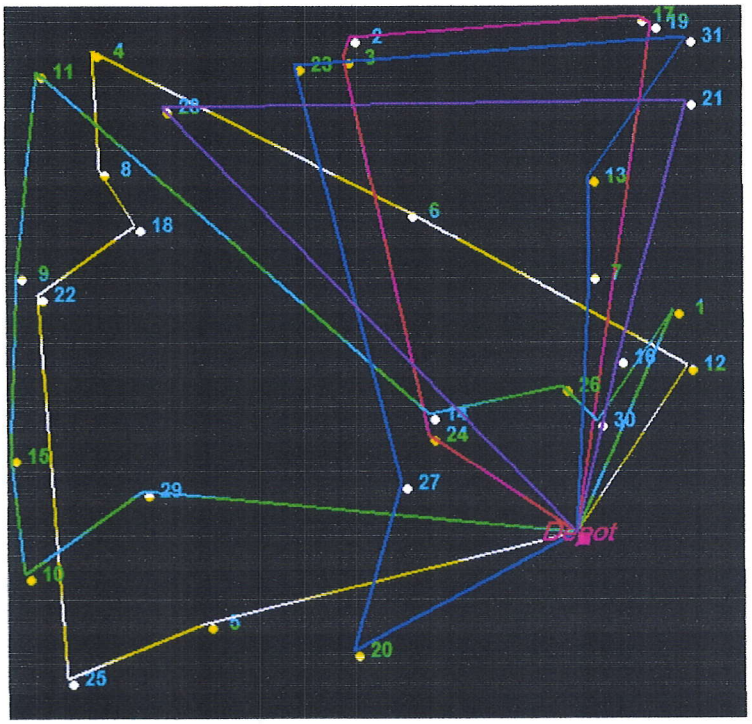


Figure 2.13 : Après l'exécution de l'algorithme 2-OPT

2.4.3 Harmony Search (Recherche harmony) HS

2.4.3.1. HS: Harmony Search (Recherche Harmony)

La recherche par harmonies (HS : Harmony Search) est une très récente métaheuristique [Amira 2013]. Elle a été proposée par Geem et ses collègues [Geem et al, 2001 ; Geem et Choi, 2007]. A l'opposé des autres métaheuristiques qui s'inspirent des

phénomènes naturels, la recherche par harmonies s'inspire du processus de recherche de la meilleure harmonie musicale. Les étapes du processus de recherche de l'algorithme de recherche par harmonies sont résumées dans l'algorithme 2.13 et expliquées dans ce qui suit [Amira 2013].

L'algorithme HS commence par une étape d'initialisation des paramètres nécessaires et de la mémoire d'harmonies (population de solution) composée d'un ensemble de 1 à HMS harmonies (i.e. solutions) aléatoires (voir figure 2.14) et des paramètres nécessaires pour le fonctionnement du HS qui sont [Amira 2013] :

- la taille de la mémoire d'harmonies (i.e. la population), notée par HMS (de l'anglais : Harmony Memory Size).

- Le taux de considération de la mémoire harmonique, noté par HMCR (de l'anglais : Harmony Memory Considering Rate), dont le rôle est de décider si la mémoire HM sera utilisée ou non.

- Le paramètre PAR (de l'anglais : Pitch Adjusting Rate), représentant la probabilité d'apporter quelques modifications à un élément de la HM.

- Le critère d'arrêt (généralement un nombre maximum d'itérations).

$$HM = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_N^1 \\ x_1^2 & x_2^2 & \dots & x_N^2 \\ \vdots & \vdots & \dots \vdots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \dots & x_N^{HMS} \end{bmatrix}$$

Figure 2.14 : La structure de la mémoire harmonique

Ensuite, l'algorithme passe à l'étape d'amélioration des harmonies. Cette étape consiste à améliorer une solution $x_i' = x_1', x_2', .. x_N'$ en se basant sur trois règles : la considération de la mémoire HM, l'ajustement des valeurs des variables de la solution et la sélection aléatoire [Amira 2013].

Après génération du nouveau vecteur (nouvelle solution) $x_i' = x_1', x_2', .. x_N'$, les composantes obtenues par considération de la mémoire HM sont examinées pour décider s'ils devront être ajustées ou non. Suit à l'étape d'amélioration de solutions, le HS passe à l'étape de mise à jour de la mémoire HM. Cette étape consiste à remplacer la mauvaise solution de la matrice HM par la nouvelle solution trouvée si cette dernière est de meilleure qualité

(comparée avec la mauvaise solution). Les étapes d'amélioration et de mise à jour seront répétées jusqu'à la satisfaction du critère d'arrêt [Amira 2013].

```
1   Debut
2       Initialiser les paramètres nécessaires ;
3       Initialiser la mémoire HM (une population d'harmonies) ;
4       Tant que la condition d'arrêt n'est pas satisfaite faire
5           Produire une nouvelle solution en améliorant la solution x'_i ;
6           Mettre à jour la mémoire HM :
7       Fin
8       Retourner la ou les meilleures solutions ;
9   Fin
```

Algorithme 2.13 : Algorithme général de la recherche par harmonies

2.4.3.2. RHD : l'algorithme de Recherche Harmonie Discret pour CVRP

Nous proposons de résoudre CVRP en utilisant l'algorithme RHD. Cette amélioration est effectuée dans le processus de génération des nouvelles harmonies.

```
    Debut
1       Réglage des paramètres : nombre d'itérations, la taille de la population HMCR, PAR.
2       Créer un ensemble initial d'harmonies satisfaisant les contraintes de problème.
3       Générer des nouvelles harmonies (approvisionnent basé sur PAR_i, HMCR_i)
4       Mise à jour de la mémoire d'harmonie
5       Répéter à partir de l'étape 3 pour l'itération suivante.
    Fin
```

Algorithme 2.14 : Algorithme RHD

2.4.3.2.1 Description de l'algorithme

Création d'une population initiale d'harmonies :

Dans la première étape, nous générons un ensemble initial d'harmonies (algorithme 2.15). Chaque harmonie code une solution possible. Chaque solution (harmonie) est générée aléatoirement (solution faisable).

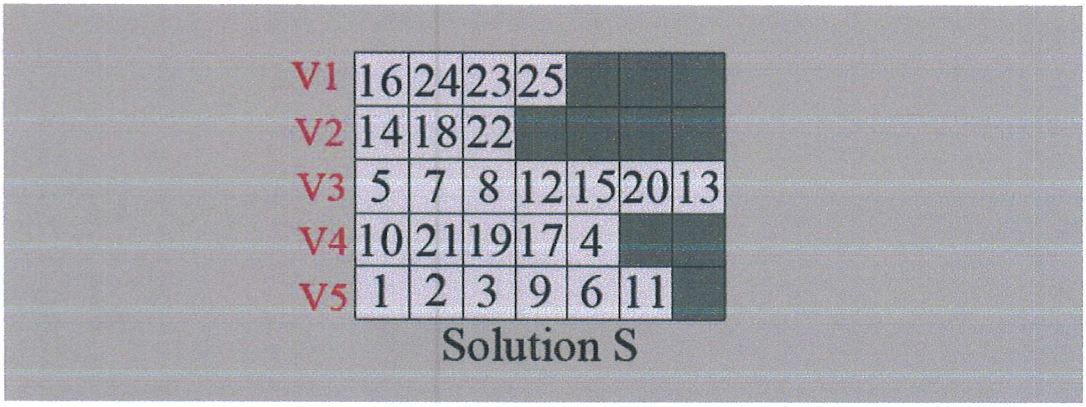


Figure 2.15 : Représentation d'une solution faisable

```
1   Debut
2       Tant que m <= nombre de population Faire
3           m := m+1
4           Trier les clients par ordre décroissant selon leurs demandes d_i
5           Répéter
6               Pour i allons de 1 à NBR_Vehicule Faire
7                   Choisir un véhicule V_i;
8                   Tant que chargeV_i <= capacity_V_i Faire
9                       Remplir V_i avec les demandes de livraisons d_i;
10                      chargeV_i := chargeV_i + d_i;
11                  Fin
12              Fin
13          Jusqùà tous les clients sont servis
14      Fin
15  Fin
```

Algorithme 2.15 : Création d'une population initiale

Génération des $HMCR_i$et PAR_i : pour chaque solution (harmonie) on génère aléatoirement un $HMCR_i$ et un PAR_i entre [0,1].

Improvisation : A chaque itération de l'algorithme, nous générons une nouvelle harmonie de deux façons possibles, en fonction de la valeur de $HMCR_i$ générée de façon aléatoire :

1. $HMCR_i$ < HMCR : Dans ce cas, la nouvelle solution est générée à partir de la mémoire d'harmonie. En effet, pour chaque x_k^{new} de l'harmonie générée (qui représente le client k à visiter), nous prenons une solution $S_t, t \in [1, HMS]$ au hasard de la mémoire puis, en fonction de la valeur du paramètre PAR, nous avons deux cas possibles :

– PAR_i< PAR : Nous faisons un ajustement de la valeur générée par le voisinage de la valeur x_k^t dans la population : soit nous prenons la valeur $x_k^{(t-1)}$ k de la solution S_{t-1} soit la valeur $x_k^{(t+1)}$k de la solutionS_{t+1}.

– PAR_i> PAR : Nous considérons la valeur x_k^t de la solution S_t. Ensuite, nous vérifions les contraintes de le la nouvelle harmonie.

2. $HMCR_i$> HMCR : Dans ce cas, et pour chaque x_k^{new} de la nouvelle harmonie, nous choisissons au hasard un client de la liste des clients disponibles. La solution produite doit satisfaire les contraintes.

2.5 Description du processus de coopération

Après avoir mis en œuvre les trois métaheuristiques présentées dans les sections précédentes, la coopération peut être développée. Par la concurrence et la coopération, cette méthode va promouvoir la diversification de même que l'intensification de l'espace de recherche. Le schéma du processus est tel que présenté dans la section 3.2. Dès qu'on

commence la recherche, les métaheuristiques esclaves ont tous les trois un rang égal à zéro. Le choix de la métaheuristique de démarrage est aléatoire. Lorsque le processus est lancé, la métaheuristique choisie aléatoirement Mènera à une solution, le gain potentiel de qualité offert par cette nouvelle solution sera ajouté au rang de métaheuristique. Les rangs des métaheuristiques esclaves aideront à terminer la prochaine métaheuristique à être exécuter. La liste taboue comprendra les métaheuristiques qui ne pourraient pas améliorer la solution actuelle si l'amélioration survient pendant l'exécution d'algorithme, la liste taboue est vidée afin que toutes les métaheuristiques esclaves soient libres à exécuter.

Le processus prend fin si les métaheuristiques n'ont pas pu améliorer la solution un nombre déterminé de fois consécutives. Une fois la liste taboue est pleine, une métaheuristique est aspirée à concurrencer les métaheuristiques en cours. Plusieurs critères d'aspiration peuvent être utilisés (la plus ancienne, la métaheuristique avec le meilleur rang ou au hasard). Nous avons opté pour un choix aléatoire entre ces options. La figure 2.16 illustre la coopération des métaheuristiques :

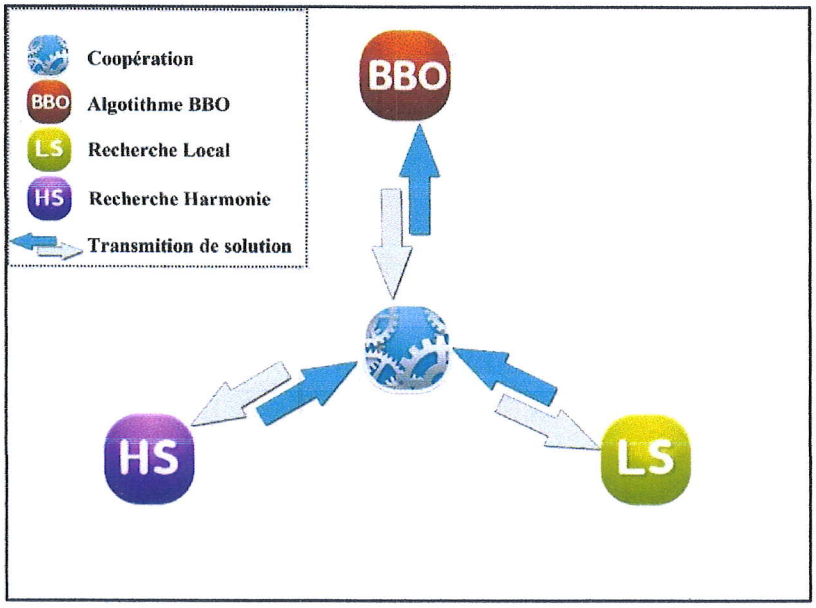


Figure 2.16 : Coopération des métaheuristiques

L'algorithme 2.16 résume le déroulement du processus de coopération.

```
1   Debut
2       Similaire := 100 ;
3       Choisir aléatoirement Meta parmi [LS ; HS ; BBO]
4       Tant que similaire > 0 faire
5           Appliquer Meta
6           Calculer amelioration_meta
7           rang_meta = amelioration_meta
8           Tant que amelioration_meta < 0 faire
9               Similaire = similaire - 1
10              Choisir Meta qui a le plus grand rang
11              Appliquer Meta
12              Calculer amelioration_meta
13              rang_meta = rang_meta + amelioration_meta
14              Ajouter Meta à liste taboue
15              Si liste pleine alors
16                  Choisir aléatoirement Meta parmi [LS ; HS ; BBO] et la libérer
17                  Sinon
18                      Choisir Meta qui a le plus grand rang
19                  Fin
20              Fin
21          Fin
22      Fin
23  Fin
```

Algorithme 2.16 : Algorithme de coopération des Métaheuristiques

Le processus de coopération est organisé comme suit :

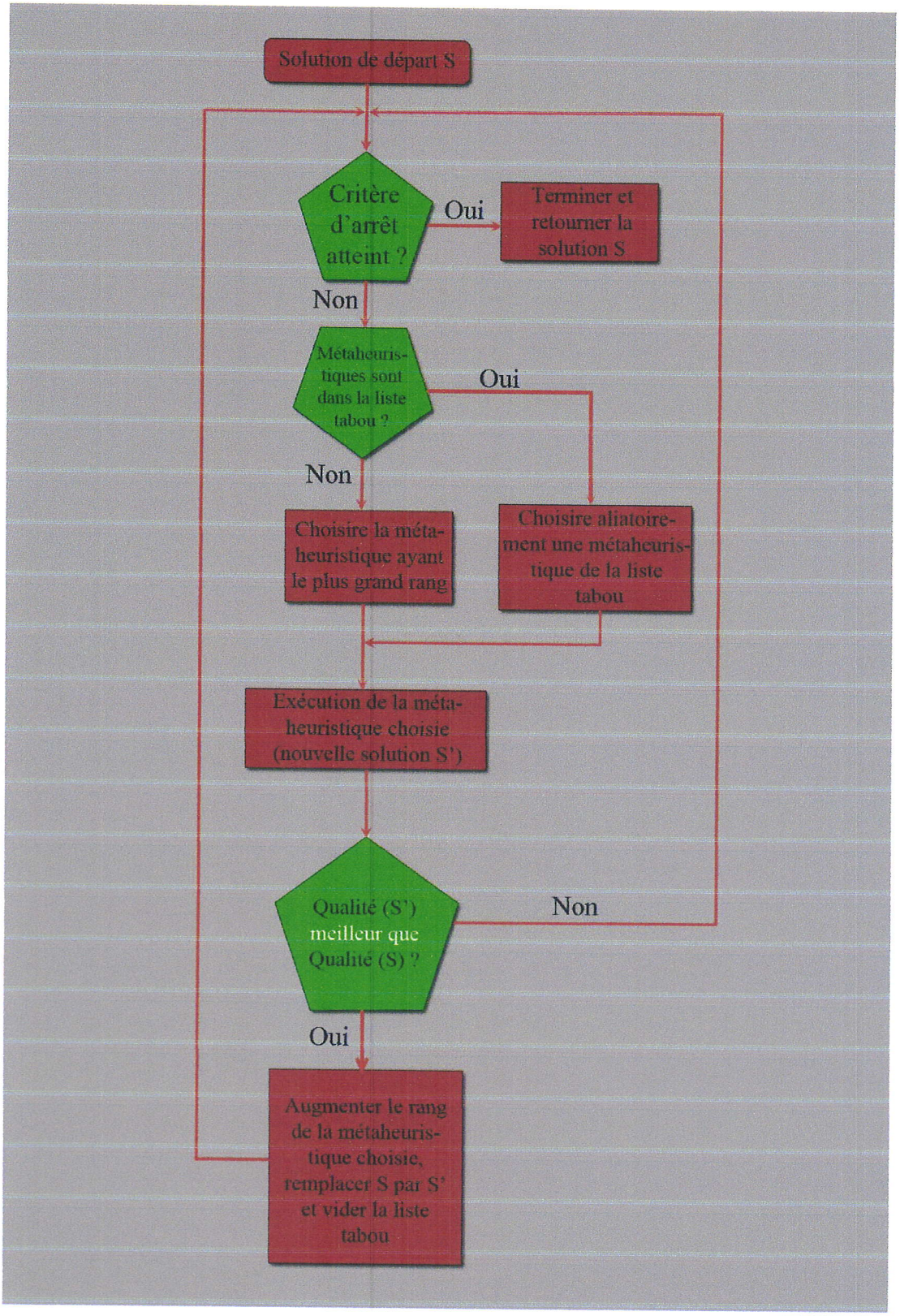


Figure 2.17 : Processus de coopération dirigée par la recherche taboue

2.6 Conclusion

Ce chapitre vise à résoudre grâce à une approche coopérative le problème de tournées de véhicules avec contrainte de capacité CVRP. L'approche proposée est une méthode de coopération entre les trois Métaheuristiques (la recherche locale multi-Start, l'algorithme de recherche harmonie et l'optimisation basée sur la biogéographie) supervisées par une Métaheuristiques maitre (recherche taboue).

Dans le chapitre qui suit, nous allons présenter nos résultats d'expérimentation.

3. Chapitre 03 : Résultats de l'approche coopérative sur le CVRP

3.1 Introduction

Dans le chapitre précédant nous avons présenté la démarche empruntée afin de réaliser un système coopératif pour la résolution du problème de tournées de véhicules avec contrainte de capacité de véhicule. Ce dernier se base sur trois métaheuristiques (Optimisation Basée sur la Biogéographie, La recherche locale multi-Start et la recherche harmonie) nommées métaheuristiques esclaves qui sont gérées par une métaheuristique maitre (la recherche taboue). Dans ce chapitre, nous présentons l'interface de l'application avec tous ces composants ainsi que les résultats d'expérimentation du processus de coopération et des différentes métaheuristiques de manière individuelle.

3.2 Présentation de l'interface

La bonne conception de l'interface graphique des applications est très importante, car elle permet de faciliter le dialogue entre l'utilisateur et le programme ainsi que d'améliorer les performances de l'application. Dans la conception de l'interface de notre application nous avons respecté un ensemble des choix ergonomiques comme la lisibilité, la compréhensibilité et l'accessibilité à tous les paramètres de l'application de manière facile et pratique,...etc. Dans ce qui suit, nous présentons des captures d'écrans des principales fenêtres de notre application.

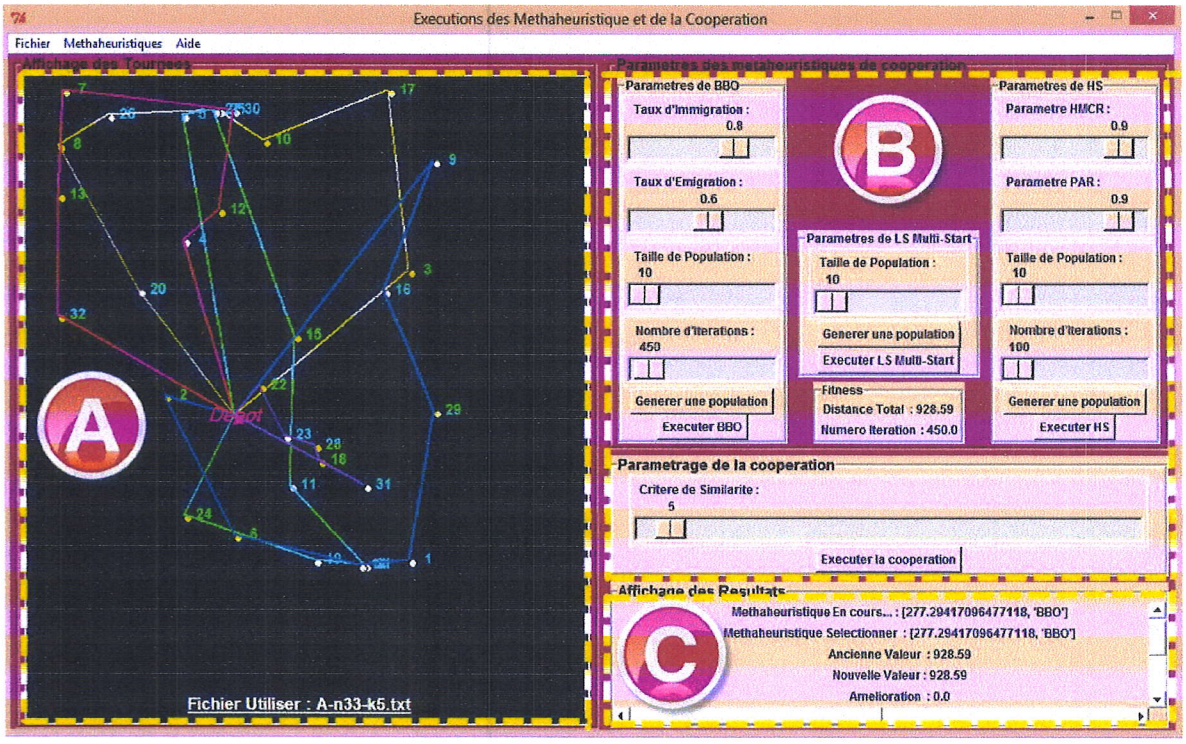


Figure 3.1: Présentation de l'interface du programme

L'interface graphique proposée dans notre application est composée de trois parties principales comme il est illustré dans la figure 3.1.

La première partie (A) est un espace pour afficher la solution (ensemble des tournées) retournée à la fin de l'exécution du programme. Les clients sont représentés par des points jaunes accompagnés du numéro de client correspondant. Les tournées sont dessinées selon l'ordre retourné par le programme où chaque tournée est représentée par une couleur différente.

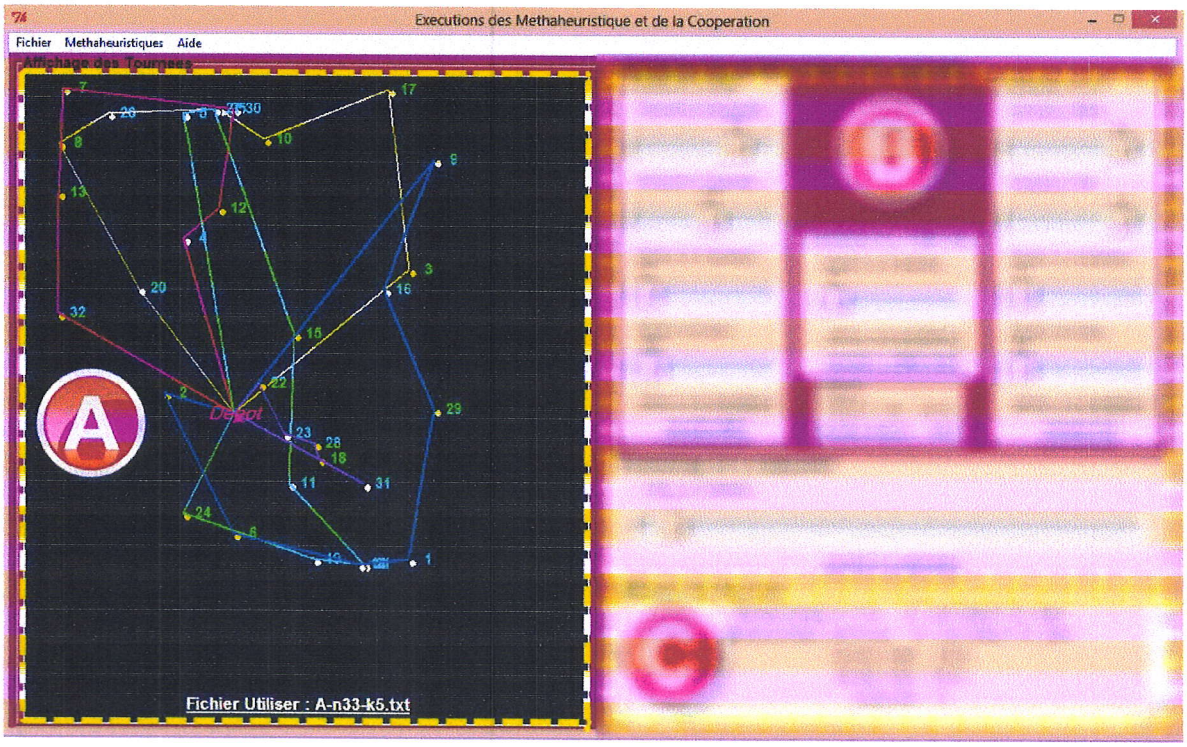


Figure 3.2 : Présentation de l'interface d'affichage des tournées

La deuxième partie (B) concerne le paramétrage des métaheuristiques et de la coopération, cette partie permet à l'utilisateur de régler les paramètres de chaque métaheuristique d'une façon individuelle. Elle permet aussi de faire une combinaison manuelle d'une coopération entre les métaheuristiques ou bien une coopération automatique comme il est illustré dans la figure 3.3.

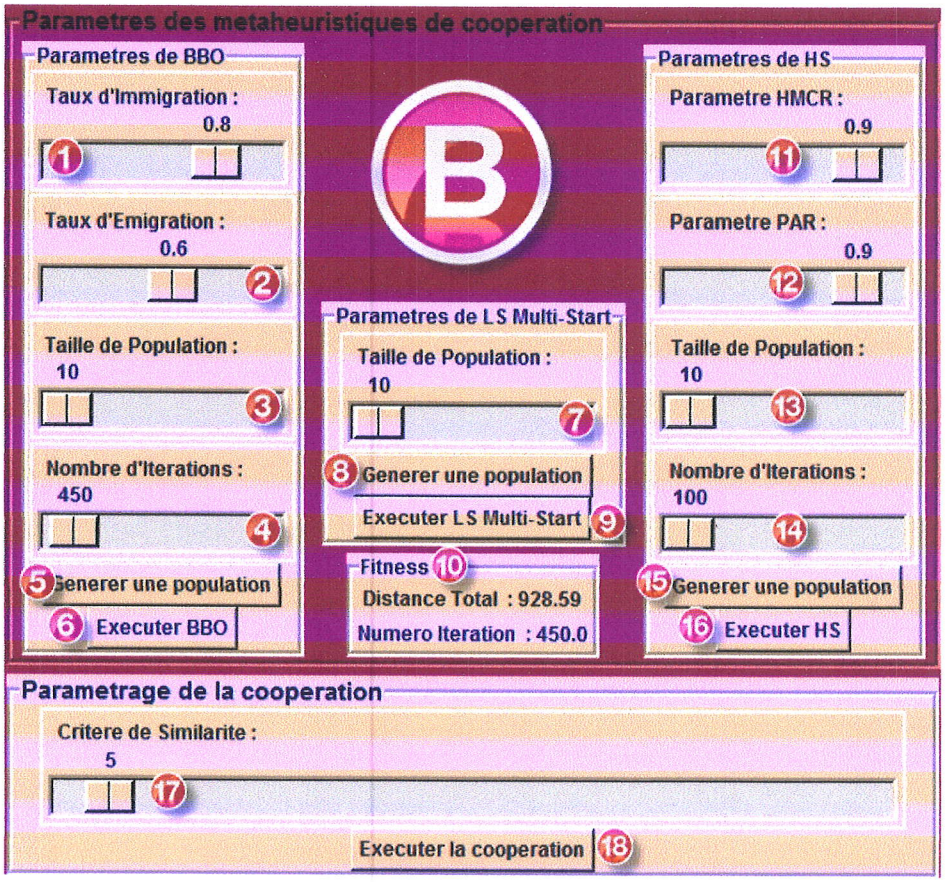


Figure 3.3 : Présentation des commandes de paramétrage des métaheuristiques

Les composants (1), (2), (3), (4), (5), (6) dans la figure 3.3 permettent de régler les paramètres : taux d'immigration, taux d'émigration, taille de la population, nombre d'itération, générer une population et exécuter l'algorithme d'optimisation Basée sur la Biogéographie respectivement. Les composants (7), (8), (9) permettent de régler la taille de la population, la générer et exécuter l'algorithme de recherche local-*Multi-Start* respectivement. Le composant (10) sert à afficher le changement du fitness de la meilleure solution quand une des métaheuristiques (ou le système coopératif) est en cours d'exécution. Les composants (11), (12), (13), (14), (15), (16) permettent de régler les paramètres : HMCR, PAR, taille de la population, nombre d'itération, générer une population et exécuter l'algorithme de recherche harmonie respectivement. Le composant (17) permet de régler le paramètre de similarité de la coopération, alors que le composant (18) permet d'exécuter cette dernière.

La troisième partie de l'interface est consacrée à l'affichage des résultats de la coopération comme le montre la figure 3.4.

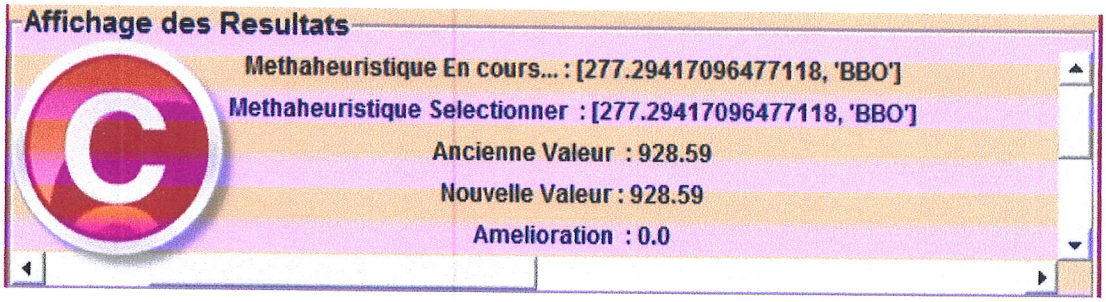


Figure 3.4 : Interface d'affichage des résultats

Le menu *Metaheuristiques* de l'application contient des boites de dialogues définissant les métaheuristiques utilisées, le processus de coopération ainsi que le problème traité dans ce travail.

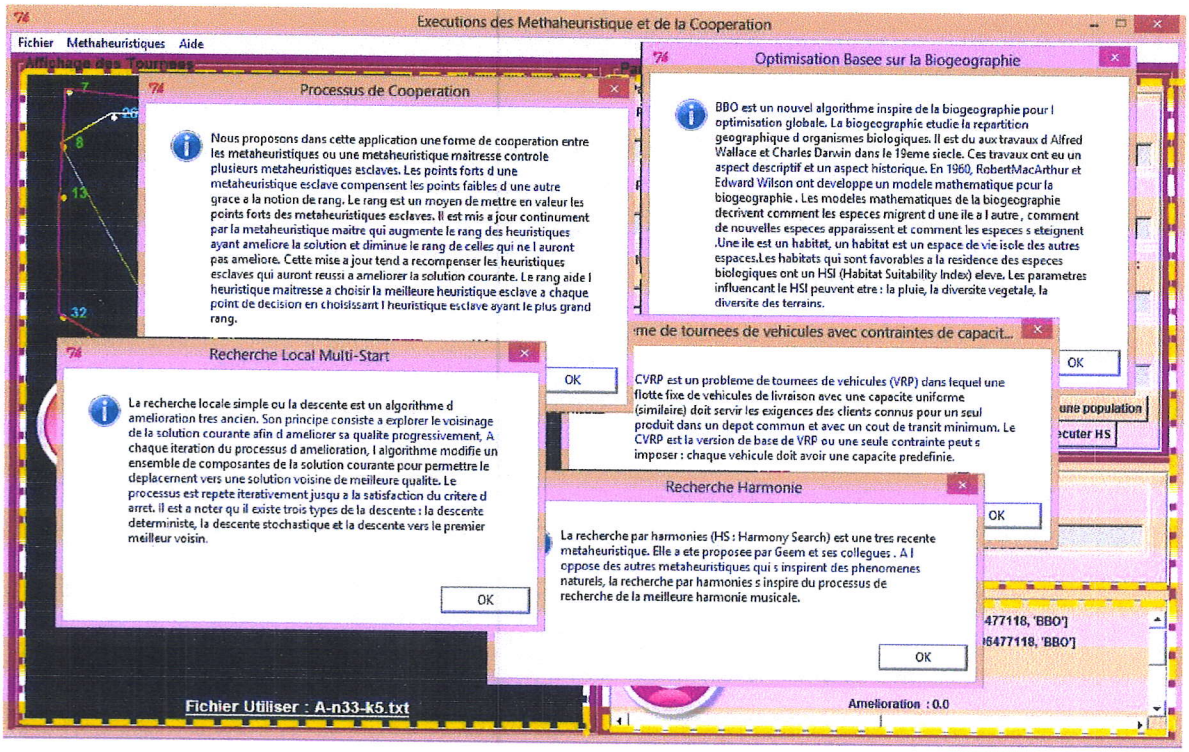


Figure 3.5: Présentation des fonctionnalités du menu Métaheuristiques

3.3 Description des Instances

Afin de tester les performances des trois métaheuristiques ainsi que le processus de coopération sur le problème de CVRP, nous avons testé nos approches sur les instances *d'Augerat et al,* ainsi que celles de *Christofides et Eilon* disponible dans la littérature.

L'ensemble des données de test est composé de 12 instances qui varient entre 16 à 39 clients et de 2 à 8 véhicules où le client numéro 0 représente le dépôt.

Exemple :

C_i	x_i	y_i	$demande_i$
0	42	68	00
1	77	97	05

Tableau 3.1 : Exemple d'instance CVRP

L'exemple précédent montre que chaque client possède une position indiquée par les coordonnées x_i et y_i et une seule demande nommée $demande_i$. Chaque fichier d'instance que nous avons utilisé est de type « **.txt* ». Il est formulé de cette manière :

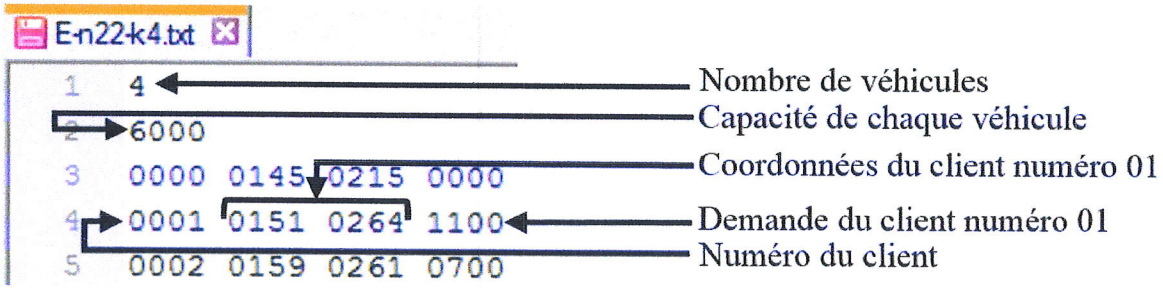


Figure 3.6 : Organisation du fichier d'instance

Il est important de noter que dans chaque fichier les deux premières lignes sont réservées. La première représente le nombre de véhicules autorisé et la deuxième représente la capacité de chaque véhicule de l'instance comme il est illustré dans la figure 3.6.

3.4 Environnement d'exécution et paramétrage

3.4.1 Environnement d'exécution

L'application est programmée en python sur l'éditeur de texte *Sublime Text*, et les tests ont été effectués sur deux ordinateurs ayant les caractéristiques suivantes :

ACER Aspire E5-573	Samsung NP300E7A
Fabricant du processeur : Intel	Fabricant du processeur : Intel
Type de processeur : Core i3	Type de processeur : Core i3
Référence du processeur : i3-4005U	Référence du processeur : i3-2330M
Vitesse du processeur : 1,70 GHz	Vitesse du processeur : 2,20 GHz
Nombre de cœur(s) du processeur : Dual-core (2-Core)	Nombre de cœur(s) du processeur : Dual-core (2-Core)
Mémoire RAM : 4 Go	Mémoire RAM : 8 Go
Technologie de la mémoire : DDR3L	Technologie de la mémoire : DDR3L
Système d'exploitation : Windows 8.1	Système d'exploitation : Windows 8

Tableau 3.2 : Caractéristiques des machines utilisées dans l'expérimentation

3.4.2 Paramétrage

Chaque métaheuristique a besoin d'un bon paramétrage au début de son exécution pour aboutir à des bons résultats. Le choix de ces paramètres est fait selon l'instance de données utilisée. Les tableaux suivants donnent les paramètres adoptés dans nos résultats pour chaque algorithme dans le processus de coopération.

	Population	Itération	Max d'espèces	Immigration	Emigration	HMCR	PAR
BBO	15 à 80	2000 à 10000	Taille de population	40 à 100 %	10 à 100 %	/	/
HS	15 à 80	100 à 2000	/	/	/	40 à 100 %	40 à 100 %
LS-multi-Start	15 à 80	/	/	/	/	/	/

Tableau 3.3 : Paramètres des métaheuristiques utilisées

Il est important de noter que le paramètre seuil de similitude de la coopération varie entre 1 jusqu'à 200 pour tous les tests effectués.

3.5 Comparaison des Résultats

Afin d'estimer l'efficacité de notre méthode de coopération ainsi que les trois métaheuristiques dans la résolution du problème de CVRP, nous avons effectué plusieurs exécutions sur les instances choisis. Les tableaux qui suivent présentent une comparaison entre les résultats obtenus par les trois métaheuristiques de manière individuelle et le processus de coopération accompagné avec des figures de courbes qui illustrent la différence entre les résultats.

Pour pouvoir mesurer la qualité des solutions de notre méthode de résolution (la coopération), on fait une comparaison de rendement avec chaque métaheuristique à savoir, Optimisation Basée sur la Biogéographie, La recherche locale multi-Start et la recherche harmonie. Ensuit on fait une comparaison de la coopération avec un Algorithme Hybride Bat avec *Path Relinking* (HBA-PR) proposés par les auteurs *Yongquan Zhou, Qifang Luo, Jian Xie* et *Hongqing Zheng* dans [Yongquan 2016] pour la résolution du problème de tournées de véhicules CVRP.

Pour cette comparaison un pourcentage d'amélioration α est calculé de la manière suivante :

$$\alpha = \left(\frac{\left| fitness_{Coop} - fitness_{metaheuristique} \right|}{fitness_{metaheuristique}} \right) * 100 \qquad (3.1)$$

A savoir aussi que si la solution ne s'améliore pas, nous mesurons l'écart β entre la solution de la coopération et celle comparée avec :

$$\beta = \left(\frac{\left| fitness_{Coop} - fitness_{BorneInf} \right|}{fitness_{BorneInf}} \right) * 100 \qquad (3.2)$$

3.5.1 Coopération avec Optimisation Basée sur la Biogéographie

Fichier utilisé	Nombre de clients	Nombre de véhicules	Coopération	BBO	α%
A-n33-k5	33	5	686,96	801,90	14,33
A-n33-k6	33	6	764,24	961,71	20,53
A-n39-k6	39	6	888,59	1 240,42	28,36
B-n35-k5	35	5	989,84	1088,58	9,07
E-n22-k4	22	4	375,27	477,10	21,34
E-n23-k3	23	3	569,75	773,86	26,38
E-n30-k3	30	3	549,45	598,96	8,27
E-n33-k4	33	4	861,13	1 031,98	16,56
P-n16-k8	16	8	454,40	486,98	6,69
P-n19-k2	19	2	221,88	243,62	8,92
P-n20-k2	20	2	221,59	255,80	13,37
P-n22-k2	22	2	222,87	261,34	14,72
Minimum					**6,69**
Maximum					**28,36**
Moyenne					**15,71**

Tableau 3.4 : Comparaison de la coopération avec BBO

Le tableau 3.4 représente les résultats comparatifs de l'approche coopérative proposée avec l'approche d'Optimisation Basée sur la Biogéographie. La première colonne représente le nom de l'instance correspondante, La deuxième et la troisième colonne représentent le nombre de clients ainsi que le nombre de véhicules respectivement, les résultats de la fonction objectif des deux approches (coopération et Optimisation Basée sur la Biogéographie) sont représentées dans la quatrième et la cinquième colonne respectivement.

A partir du tableau 3.4, on remarque que l'approche coopérative a améliorée les résultats retournés par BBO avec un pourcentage qui varie entre **6,69%** et **28,36%**. Le graphe suivant représente la différence entre les résultats retournés par l'approche de coopération et qui sont retournés par BBO.

	A-n33-k5	A-n33-k6	A-n39-k6	B-n35-k5	E-n22-k4	E-n23-k3	E-n30-k3	E-n33-k4	P-n16-k8	P-n19-k2	P-n20-k2	P-n22-k2
Coopération	686,96	764,24	888,59	989,84	375,27	569,75	549,45	861,13	454,4	221,88	221,59	222,87
BBO	801,9	961,71	1 240,4	1088,58	477,1	773,86	598,96	1 031,9	486,98	243,62	255,8	261,34

Figure 3.7 : Graphe des résultats de BBO et la coopération

3.5.2 Coopération avec recherche locale *multi-start*

Fichier utilisé	Nombre de clients	Nombre de véhicules	Coopération	LS-MS	α%
A-n33-k5	33	5	686,96	1099,93	37,55
A-n33-k6	33	6	764,24	1193,59	35,97
A-n39-k6	39	6	888,59	1527,48	41,83
B-n35-k5	35	5	989,84	1551,86	36,22
E-n22-k4	22	4	375,27	620,47	39,52
E-n23-k3	23	3	569,75	719,95	20,86
E-n30-k3	30	3	549,45	802,09	31,50
E-n33-k4	33	4	861,13	1109,32	22,37
P-n16-k8	16	8	454,40	473,74	4,08
P-n19-k2	19	2	221,88	235,50	5,78
P-n20-k2	20	2	221,59	279,75	20,79
P-n22-k2	22	2	222,87	298,11	25,24
		Minimum			**4,08**
		Maximum			**41,83**
		Moyenne			**26,81**

Tableau 3.5 : Comparaison de la coopération avec la recherche locale multi-start

Le tableau 3.5 représente les résultats comparatifs de l'approche coopérative proposée avec l'approche de la recherche local *Multi-Start*. On remarque que l'approche de coopération a améliorée les résultats de la recherche locale *Multi-Start* avec un pourcentage qui varie entre **4,08%** et **41,83%**. Le graphe qui suit représente la différence entre les résultats des deux approches (coopération, recherche locale *Multi-start*).

	A-n33-k5	A-n33-k6	A-n39-k6	B-n35-k5	E-n22-k4	E-n23-k3	E-n30-k3	E-n33-k4	P-n16-k8	P-n19-k2	P-n20-k2	P-n22-k2
Coopération	686,96	764,24	888,59	989,84	375,27	569,75	549,45	861,13	454,4	221,88	221,59	222,87
LS-Multi-Start	1099,93	1193,59	1527,48	1551,86	620,47	719,95	802,09	1109,32	473,74	235,5	279,75	298,11

Figure 3.8 : Graphe des résultats de la recherche locale Multi-Start et la Coopération

3.5.3 Coopération avec recherche harmonie

Le tableau 3.6 représente les résultats comparatifs de l'approche coopérative proposée avec l'approche de la recherche harmonie. On remarque que l'approche de coopération a améliorée les résultats de la recherche harmonie avec un pourcentage qui varie entre **8,08%** et **55,48%**. Le graphe dans la figure 3.9 représente la différence entre les résultats des deux approches (coopération, recherche harmonie).

Fichier utilisé	Nombre de clients	Nombre de véhicules	Coopération	HS	α%
A-n33-k5	33	5	686,96	1224,29	43,89
A-n33-k6	33	6	764,24	1292,93	40,89
A-n39-k6	39	6	888,59	1870,05	52,48
B-n35-k5	35	5	989,84	1893,55	47,73
E-n22-k4	22	4	375,27	591,62	36,57
E-n23-k3	23	3	569,75	1013,19	43,77
E-n30-n3	30	3	549,45	1234,05	55,48
E-n33-k4	33	4	861,13	1280,98	32,78
P-n16-k8	16	8	454,40	494,32	8,08
P-n19-k2	19	2	221,88	342,37	35,19
P-n20-k2	20	2	221,59	383,19	42,17
P-n22-k2	22	2	222,87	382,45	41,73
Minimum					**8,08**
Maximum					**55,48**
Moyenne					**40,06**

Tableau 3.6 : Comparaison de la coopération avec la recherche harmonie

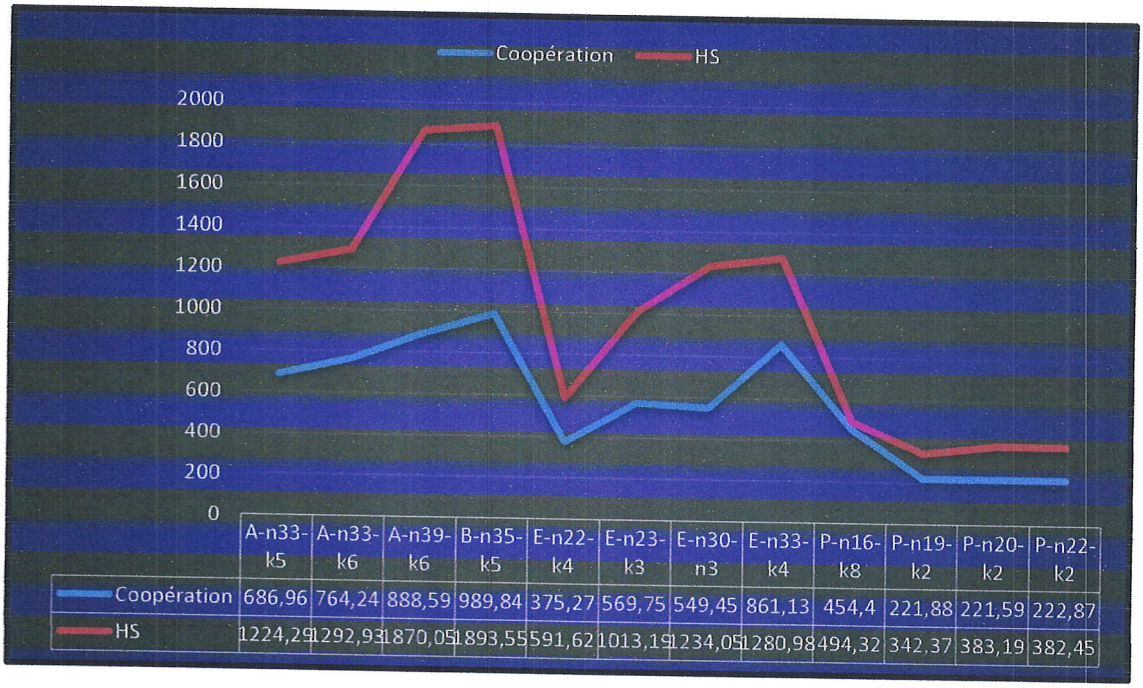


Figure 3.9 : Graphe des résultats de la recherche harmonie et la coopération

Selon les trois tableaux 3.4, 3.5 et 3.6 on remarque que l'approche de coopération a améliorée les résultats des trois métaheuristiques (Optimisation Basée sur la Biogéographie,

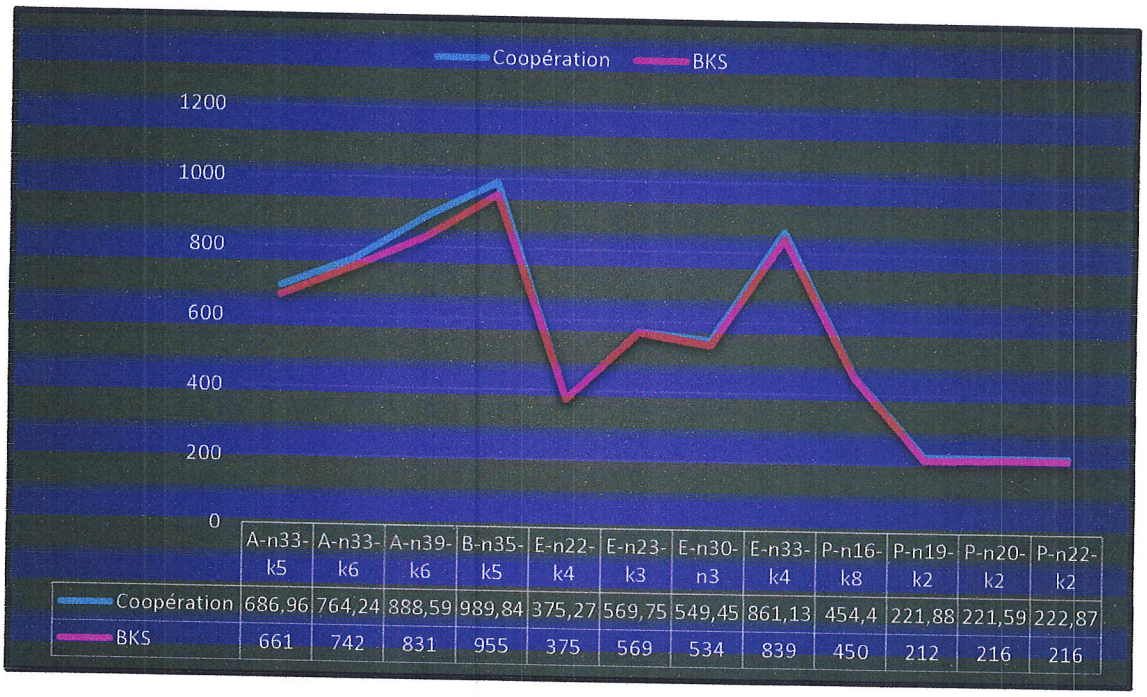
La recherche locale multi-Start et la recherche harmonie) avec des moyennes de **(15,71%, 26,81%, 40,06%)** respectivement, ce qui signifie que l'approche d'Optimisation Basée sur la Biogéographie est plus efficace que celle de la recherche locale multi-Start et la recherche harmonie.

3.5.4 Coopération avec BKS (Best Known Solutions)

Fichier utilisé	Nombre de clients	Nombre de véhicules	Coopération	BKS	β%
A-n33-k5	33	5	686,96	661	3,78
A-n33-k6	33	6	764,24	742	**2,91**
A-n39-k6	39	6	888,59	831	6,48
B-n35-k5	35	5	989,84	955	3,52
E-n22-k4	22	4	375,27	375	**0,07**
E-n23-k3	23	3	569,75	569	**0,13**
E-n30-k3	30	3	549,45	534	**2,81**
E-n33-k4	33	4	861,13	839	**2,57**
P-n16-k8	16	8	454,40	450	**0,97**
P-n19-k2	19	2	221,88	212	4,45
P-n20-k2	20	2	221,59	216	**2,52**
P-n22-k2	22	2	222,87	216	3,08
Minimum					**0,07**
Maximum					**6,48**
Moyenne					**2,78**

Tableau 3.7 : Comparaison de la coopération avec BKS

Le Tableau 3.7 représente la comparaison de la qualité des solutions entre l'approche de coopération et BKS. On remarque une insuffisance de performance de l'approche de coopération. L'écart entre les deux résultats est minimisé (en moyenne **2,78%**) pour la majorité des instances. Le plus grand écart trouvé est de **6,48%** dans l'instance **A-n39-k6**. Le graphe suivant représente la différence entre les deux résultats.



	A-n33-k5	A-n33-k6	A-n39-k6	B-n35-k5	E-n22-k4	E-n23-k3	E-n30-n3	E-n33-k4	P-n16-k8	P-n19-k2	P-n20-k2	P-n22-k2
Coopération	686,96	764,24	888,59	989,84	375,27	569,75	549,45	861,13	454,4	221,88	221,59	222,87
BKS	661	742	831	955	375	569	534	839	450	212	216	216

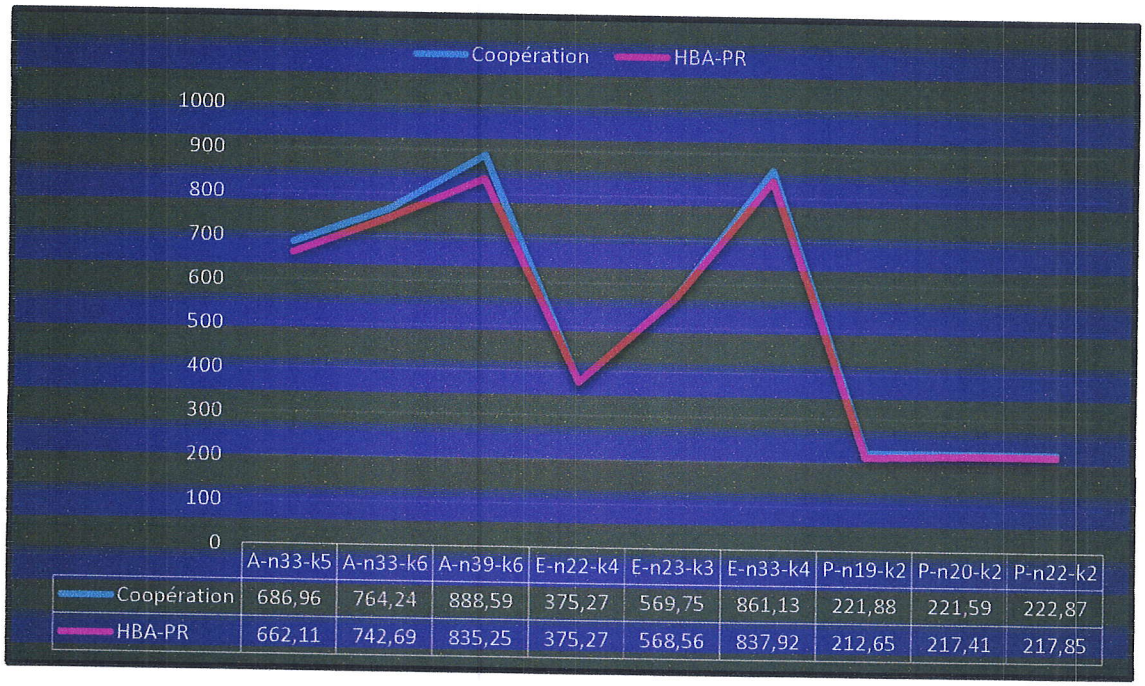
Figure 3.10 : Graphe des résultats de BKS et la Coopération

3.5.5 Coopération avec HBA-PR

Fichier utilisé	Nombre de clients	Nombre de véhicules	Coopération	HBA-PR	β%
A-n33-k5	33	5	686,96	662,11	3,62
A-n33-k6	33	6	764,24	742,69	**2,82**
A-n39-k6	39	6	888,59	835,25	6,00
B-n35-k5	35	5	989,84	-	-
E-n22-k4	22	4	375,27	375,27	**0,00**
E-n23-k3	23	3	569,75	568,56	**0,21**
E-n30-k3	30	3	549,45	-	-
E-n33-k4	33	4	861,13	837,92	**2,70**
P-n16-k8	16	8	454,40	-	-
P-n19-k2	19	2	221,88	212,65	4,16
P-n20-k2	20	2	221,59	217,41	**1,89**
P-n22-k2	22	2	222,87	217,85	**2,25**
			Minimum		**0,00**
			Maximum		**6,00**
			Moyenne		**2,63**

Tableau 3.8 : Comparaison de la coopération avec HBA-PR

Le Tableau 3.8 représente la comparaison de la qualité des solutions entre l'approche de coopération et l'approche de HBA-PR. On remarque aussi une insuffisance de performance de l'approche de coopération. L'écart entre les deux résultats est minimisé (en moyenne **2,63%**) pour la majorité des instances. Le plus grand écart trouvé est de **6,00%** dans l'instance **A-n39-k6**. Le graphe suivant représente la différence entre les deux résultats.



	A-n33-k5	A-n33-k6	A-n39-k6	E-n22-k4	E-n23-k3	E-n33-k4	P-n19-k2	P-n20-k2	P-n22-k2
Coopération	686,96	764,24	888,59	375,27	569,75	861,13	221,88	221,59	222,87
HBA-PR	662,11	742,69	835,25	375,27	568,56	837,92	212,65	217,41	217,85

Figure 3.11 : Graphe des résultats de l'approche HBA-PR et la Coopération

On remarque dans le tableau 3.8 que l'écart reste proportionnel à la grandeur des instances. En effet, la différence est acceptable quand il s'agit des petites instances, et augmente relativement pour les grandes.

3.5 Conclusion :

L'approche de coopération exploite les avantages de chaque métaheuristique pour diversifier mieux l'espace de recherche, elle retourne des résultats relativement plus performants. Cependant, il reste à effectuer des expérimentations plus profondes afin de parvenir à de meilleurs résultats ainsi que de confirmer l'importance de l'approche coopérative.

4. Chapitre 04 : Conclusion et Perspectives

Ce chapitre clôt le mémoire en récapitulant les objectifs et les points abordés. Ce mémoire avait pour objet la résolution grâce à une méthode coopérative, l'une des variantes du problème de tournées de véhicules, qui malgré son ancienneté, reste l'un des problèmes les plus traités dans le domaine de l'optimisation combinatoire.

Les problèmes de tournées de véhicules (Vehicle Routing Problems) sont une grande famille de problèmes d'optimisation combinatoire, depuis sa mise en œuvre par Dantzig et Ramser il y a plus de 50 ans sous le titre de « The Truck Dispatching Problem » [Dantzig 1959], des centaines d'articles ont été consacrées à sa résolution exacte ou approximative.

Dans ce mémoire, nous avons étudié une variante du VRP nommée le CVRP (Capacitated Vehicle Routing Problem) formulée en 1976 par [Christofides 1976], dans lequel une flotte fixée de véhicules de livraison avec une capacité uniforme (similaire) doit servir les exigences des clients connus pour un seul produit dans un dépôt commun et avec un coût de transit minimum.

Après avoir présenté une description du problème de VRP nous avons cité certaines variantes de ce dernier et leurs formulations mathématiques.

La résolution des modèles mathématiques du problème et de ces multiples variantes, est réalisée par de nombreuses méthodes, chacune fournissant des approches de recherche différentes, la partie qui a suivi abordait, de façon théorique, les principes de fonctionnements des méthodes de résolution utilisées pour la réalisation de ce travail.

L'approche proposée pour la résolution du problème étudié (CVRP) est une méthode de coopération entre des métaheuristiques gérées par une métaheuristique maitre, l'optimisation basée sur la biogéographie, la recherche locale Multi-Start et la recherche harmonie, ces dernières ont pour principale tâche de fusionner leurs fonctionnement sous le contrôle de la métaheuristique maitre afin d'exploiter les points forts de chaque métaheuristique et d'atteindre une qualité de solution meilleure.

Afin d'implémenter les différentes métaheuristiques, il a fallu adapter chacune d'elle de manière adéquate au problème traité, nous avons présenté chaque opérateur de modification ou de mise à jour des solutions.

Enfin un chapitre d'observation et d'expérimentation vient conclure ce travail et expose la qualité et l'efficacité de l'approche proposée, tous cela en effectuant des différentes comparaisons avec les résultats retournés par les exécutions des métaheuristiques de manière individuelle puis avec quelques résultats de la littérature qui traitent le même problème.

L'expérimentation de cette méthode coopérative a montrée dans tous les cas, une amélioration de qualité. En effet cette fusion du fonctionnement de ces métaheuristiques a profitée des avantages de chacune d'elles. L'espace de recherche est exploré de manière plus approfondit à chaque exécution d'une de ces métaheuristiques et les solutions de départ sont de plus en plus améliorées, bien que dans la plupart des cas le réglage des paramètres n'étaient pas efficace au maximum à cause de la contrainte de temps qui est fixé pour la réalisation de ce travail.

On souhaite désormais d'établir des analyses expérimentales plus poussées, afin de valider l'efficacité de cette méthode coopérative de manière définitive.

Bibliographie

[Amira 2013] Amira Gherboudj. *Méthodes de résolution de problèmes difficiles académiques*. Thèse de doctorat 3ème cycle LMD Spécialité : Informatique. Sous la direction de SALIM CHIKHI. Université de Constantine2. 216 pages, 2013.

[Berghida 2015] Berghida Meryem. *Stratégies adaptatives et coopératives pour la résolution de problèmes de tournées de véhicules avec collecte et livraison*. Thèse de doctorat en informatique Spécialité : Systèmes Informatiques. Sous la direction de Boukra Abdelmadjid. Université des Sciences et de la Technologie Houari Boumediene. 121 pages, 2015.

[Christofides 1976] Nicos Christofides. *The vehicle routing problem*. RAIRO - Operations Research - Recherche Opérationnelle, vol. 10, no. V1, pages 55-70, 1976.

[Cordeau 2001] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon et F. Soumis. *The Vehicle Routing Problem*. Chapitre VRP with Time Windows. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pages 157-193, 2001.

[Chen 2012a] R.-M. Chen, F.-R. Hsieh et D.-S. Wu. *Heuristics based ant colony optimization for vehicle routing problem*. In Proceedings of the 2012 7th IEEE Conference on Industrial Electronics and Applications, ICIEA 2012, pages 1039-1043, 2012.

[Coo 2014] Cooperative Coevolution. In Support Vector Machines and Evolutionary Algorithms for Classification SE - 5, volume 69 of Intelligent Systems Reference Library. Springer International Publishing, pages 57-73 2014.

[Croes 1958] G. A. CROES. *A method for solving traveling salesman problems*. Operations Res. Pages 791-812. 6, 1958.

[Dantzig 1959] G B Dantzig et J H Ramser. *The Truck Dispatching Problem*. Management Science, vol. 6, pages 80-91, Octobre 1959.

[Desaulniers 2001] G. Desaulniers, J. Desrosiers, A. Erdmann, M. M. Solomon et F. Soumis. *The Vehicle Routing Problem. chapitre VRP with Pickup and Delivery*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pages 225-242, 2001.

[Darwin 1995] Charles Darwin. *The origin of species*. New York: P.F. Collier↕. http://www.biodiversitylibrary.org/bibliography/24252. 1995.

[Fleurent 1994] Charles Fleurent et Jacques A Ferland. *Algorithmes Géné-tiques Hybrides Pour L'optimisation Combinatoire*, 1994.

[Geem et al 2001] Z. W. Geem, J. H. Kim and G. V. *Loganathan. A New Heuristic Optimization Algorithm : Harmony Search. Simulation*. Vol. 76, N° 2, pages 60-68, 2001.

[Geem et Choi 2007] Z. W. Geem and J.Y Choi. *Music composition Using Harmony Search Algorithm*. M. Giacobini et al. (Eds.) : EvoWorkshops 2007, LNCS 4448, Springer, pages 593-600, 2007.

[HAJ 2010] Mais HAJ RACHID. *Les problèmes de tournées de véhicules en planification industrielle : classification et comparaison d'opérateurs évolutionnaires*. Thèse de doctorat en informatique. Sous la direction de François SPIES, Christelle BLOCH, Pascal CHATONNAY, Wahiba RAMDANE-CHERIF. Université de Franche-Comté. 277 pages, 2010.

[Laporte 1992] G. Laporte, *The vehicle routing problem: An overview of exact and approximate algorithms*, Eur. J. Oper. Res., vol. 59, no. 3, pages 345-358, 1992.

[MacArthur 1967] R.H. MacArthur ET E.O. WILSON. *The theory of island biogeography*. Monographs in population biology. Princeton University Press, 1967.

[Naimi 2008] M. Naimi. *Amélioration de la Performance des Algorithmes Evolutionnaires : Multiobjectifs : Application au Problème de Sac à Dos Multidimensionnel Multiobjectif*. Thèse de doctorat. Université Hassan 2 Casablanca, Maroc. 2008.

[Potter 1994] Mitchell A Potter et Kenneth A De Jong. *A Cooperative Coevolutionary Approach to Function Optimization*. In Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: Parallel Problem Solving from Nature, PPSN III, London, UK, UK. Springer-Verlag, Pages 249-257 ,1994.

[Preux 1999] P Preux et E.-G. Talbi. *Towards hybrid evolutionary algorithms*. International Transactions in Operational Research, vol. 6, no. 6, pages 557-570, 1999.

[Rego 1994] C Rego, C Roucairol et Institut National de recherche en informatique et en automatique. *Le problème de tournées de véhicules : étude et résolution approchée*. Rapports de recherche. Institut national de recherche en informatique et en automatique, 1994.

[Ralphs 2003] Ted K. Ralphs, L. Kopman, William R. Pulleyblank et Leslie E. Trotter Jr. *On the capacitated vehicle routing problem*. Math. Program., vol. 94, no. 2-3, pages 343-359, 2003.

[Ralphs, 2003] T. K. Ralphs. *Parallel branch and cut for capacitated vehicle routing*. Parallel Computing archive, 29 (5), pages 607-629, 2003.

[Sariklis 2000] D Sariklis et S Powell. *A heuristic method for the open vehicle routing problem*. Journal of the Operational Research Society, vol. 51, no. 5, pages 564-573, 2000.

[Sahbi 2011] Sahbi Ben Ismail, François Legras, Gilles Coppin. *Synthèse du problème de routage de véhicules*. Rapport de recherche Dépt. Logique des Usages, Sciences Sociales et de l'Information (Institut Mines-Télécom-Télécom Bretagne-UEB) ; Laboratoire en sciences et technologies de l'information, de la communication et de la connaissance (UMR CNRS 6285 - Télécom Bretagne - Université de Bretagne Occidentale - Université de Bretagne Sud). Page 52, 2011.

[Simon 2008] Dan Simon. *Biogeography-Based Optimization*. IEEE Trans. Evolutionary Computation, vol. 12, no. 6, pages 702-713, 2008.

[Toth 2001] Paolo Toth et Daniele Vigo, editeurs. *The vehicle routing problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[Wolpert 1997] David H. Wolpert et William G. Macready. *No free lunch theorems for optimization*. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, vol. 1, no. 1, pages 67-82, 1997.

[Wallace 1876a] Alfred Russel Wallace. *The geographical distribution of animals*. Volume v.1. New York,Harper and brothers↕. http://www.biodiversitylibrary.org/bibliography/11354. 1876.

[Wallace 1876b] Alfred Russel Wallace. *The geographical distribution of animals*. Volume v.2. New York,Harper and brothers↕. http://www.biodiversitylibrary.org/bibliography/11354. 1876.

[Yongquan 2016] Y. Zhou, Q. Luo, J. Xie, and H. Zheng, *A hybrid bat algorithm with path relinking for the capacitated vehicle routing problem*, Model. Optim. Sci. Technol., vol. 7, pages 255–276, 2016.

[Zhu, 1999] K.Q. Zhu. *Heuristic Methods for Vehicle Routing Problem with Time Windows*. PhD thesis, National University of Singapore, Departement of Electrical Engineering, May, 1999.

[Zhengchu 2010] Wang Zhengchu, Zhou Muxun, Li Xiufeng, Fan Chun et Jin Feixiang. *A quantum particle swarm optimization for solving the capacitated vehicle routing problem*. In Intelligent Control and Automation (WCICA), 2010 8th World Congress on, pages 3281-3285, 2010.

Résumé

Ce sujet situant dans le cadre d'optimisation combinatoire, a l'objectif de résoudre le problème VRP avec une approche coopérative de métaheuristiques relativement récentes.

Le problème de distribution des biens dans lequel les véhicules basés à un dépôt central sont tenus de visiter (pendant une période de temps) des clients géographiquement dispersés afin de satisfaire leurs exigences connues est appelé le VRP (Vehicle Routing Problem). Le problème de tournées de véhicules (VRP) est l'un des plus célèbres problèmes d'optimisation combinatoire, étant considéré comme une extension de TSP, il est donc un problème NP-Difficile.

L'idée de coopérer les algorithmes de recherche est classique en optimisation combinatoire. Elle peut prendre la forme de co-évolution coopérative CC, algorithmes hybrides (hybride séquentiel, hybride parallèle synchrone, hybride parallèle asynchrone) ou les hyperheuristiques. Nous proposons dans ce travail une autre forme de coopération entre les métaheuristiques où une métaheuristique maitre contrôle plusieurs métaheuristiques esclaves pour résoudre le problème de tournées de véhicules VRP).

Mots clés : Problème de tournées de véhicules, CVRP, Optimisation basée sur la biogéographie, recherche d'harmonie, recherche locale, MLS, Recherche taboue, métaheuristiques, Coopération.

Abstract

This subject, located within combinatorial optimization, has the aim of solving the VRP problem with a cooperative approach of relatively recent metaheuristics.

The problem of distribution of goods in which the vehicles based at a central depot are required to visit (for a period of time) a set of customers geographically dispersed in order to satisfy their demands is called VRP (Vehicle Routing Problem). The VRP is one of the most famous combinatorial optimization problems, is considered as an extension of TSP, so it is an NP-hard problem.

The idea of cooperating search algorithms is classic in combinatorial optimization. It can take the form of cooperative co-evolution CC, hybrid algorithms (sequential hybrid, hybrid synchronous parallel, asynchronous parallel hybrid) or hyperheuristics. We propose in this work a different form of cooperation between metaheuristics, where a master metaheuristic controls several slaves' metaheuristics to solve the VRP problem.

Keywords: Capacitated Vehicle Routing Problem, Biogeography Based Optimization, Harmony Search, Local Search, MLS, Tabu Search, Cooperation, metaheuristic.