République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de SAAD DAHLAB Blida

USDB

Faculté des sciences

Département informatique

Mémoire pour l'obtention D'un diplôme de Master

en Génie Logiciel

THÈME

Implémentation d'un algorithme intelligent pour la classification non supervisée partielle

Présenté par: Lamali Ahlem

Messak Hafsa

Promoteur : Dr.Djenouri Djamel

Encadreur : Mr .Djenouri Youcef

Promotion 2012/2013

Remerciements

Nous remercions avant tout le bon dieu qui nous a aidés à réaliser ce modeste travail.

Nous tenons à remercier notre promoteur Dr. Djenouri Djamel pour sa patience, sa disponibilité, et sa compréhensibilité.

Nous remercions les membres du jury pour nous avoir fait l'honneur de juger notre travail.

Nous tenons à remercier notre encadreur Mr. Djenouri Youcef pour son aide, sa disponibilité.

Nous remercions tous les enseignants de la faculté des sciences de BLIDA, et particulièrement nos enseignants du département informatique.

Par ailleurs, Nous remercions de tout cœur, tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail.

Dédicace

Ce mémoire est dédié :

A la mémoire de mon Père Rabah et à tous ceux qui nous ont quittés.

A La femme qui a donnée de son mieux pour moi et qui a longtemps attendu ce jour ma chère Maman que Dieu la protège.

Comme je réserve un merci particulier à ma grande mère Fatima El Zohra que Dieu la protège ainsi mon oncle SILMI Ahmed.

A mes frères : Djamel, Abd El-Ali, Abd El-Wahab.

A la fleur de notre maison ma petite sœur Maroua.

A la familles LAMALI et SILMI.

A mon binôme Messak Hafsa je lui souhaite une vie pleine de bonheur.

A mes cousines sans exception.

A tous les étudiants de Master 2 surtout sofiane et Dahman Souad.

A tous qui me souhaitent du bien.

Ahlem...

Dédicace

Ce mémoire est dédié :

A mes très chers parents pour leur soutient durant toute ma période d'étude, pour leur

bienveillance, leurs efforts constants, et leurs encouragements.

A mes grands-parents que Dieu les protègent.

A tous mes frères : Samir, Zinou, Amine, Hamza, Ismail.

A mes sœurs : Chahrazed, et ma petite sublime Kaouther.

A mon fiancé Zakaria

A la familles MESSAK, BEN ABED et TETBIRT.

A mon binôme Lamali Ahlem.

A mes meilleures amis Ahlem, sofiane et tous les étudiants de Master 2.

A tous qui me souhaitent du bien.

**** Hafsa****

Résumé

K-means est l'un des algorithmes qui résout le problème de classification bien connue, mais il est sensible aux centres initiaux des classes.

La colonie des abeilles est un algorithme d'optimisation stochastique P-métaheuristique qui appartient à l'algorithme essaim d'intelligence, dans les dernières décennies, plusieurs études basées sur différent comportements de colonie d'abeilles sont développées pour résoudre les problèmes complexes d'optimisation combinatoire.

Pour cela, dans ce présent travail, on va proposer une hybridation entre K-means et l'algorithme des abeilles (BSO) afin de régler le problème d'initialisation de K-means.

Mots clés

La fouille de données, K-means, métaheuristique et BSO.

Abstract

K-means is one of well-known algorithm that solves the clustering problem, but it is sensitive to the initial cluster centers.

The bee colony algorithm is a stochastic optimization P-metaheuristic algorithm which belongs to the swarm intelligence, in the last decades, several studies based on different behaviors bee colony have been developed to deal the combinatorial optimization problems.

For this, in this work, a new intelligent algorithm is proposed which is grounded on both K-means and bees behaviors. The results are very promising, furthermore the proposed algorithm allow us to solve the problem of K-means mentioned above.

Keywords

Data mining, k-means, metaheuristic and BSO.

ملخص

k-means هو واحد من الخوارزميا ت التي تحل مشكلة التصنيف المعروفة ،ولكنه حساس للمراكز الأساسية.

خوارزمية مستعمرة النحل تنتمي إلى زمرة الخوارزميات الذكية. في العقود الأخيرة، وضعت العديد من الدراسات على أسس مختلفة لسلوكيات مستعمرة النحل من أجل حل المشاكل.

لهذا، في هذا العمل، اقترحنا خوارزمية ذكية جديدة ترتكز على كل من k-means و سلوكيات النحل. و كانت النتائج واعدة جدا، وعلاوة على ذلك الخوارزمية المقترحة تسمح لنا لحل مشكلة k-means المذكورة أعلاه.

الكلمات الرئيسية

استخراج البيانات،k-means ، métaheuristique وBSO.

TABLE DE MATIERE

TABLE DE MATIERE

Chapitre II : Les métaheuristiques

TABLE DE MATIERE

Chapitre III : La classification non supervisée

TABLE DE MATIERE

Chapitre IV : La modélisation d'un algorithme intelligent

TABLE DE MATIERE

Chapitre V : Implémentation et expérimentation

Liste des figures

Liste des figures

Liste des Tableaux

Liste des Acronymes

KDD	*Knowledge Discovery in Data bases*
CHA	*Classification Hiérarchique Ascendante*
KM	*K-Means*
KPPV	*K - Plus Proche Voisin*
FIFO	*First In First Out*
GA	*Genetic Algorithm*
PSO	*Particle Swarm Optimization*
TS	*Tabu Search*
GAIK	*Genetic Algorithm Initializes K-Means*
KIGA	*K-means Initializes Genetic Algorithm*
HGAPSOA	*Hybrid of Genetic Algorithm and Particle Swarm Optimization Algorithm*
BSO	*Bees Swarm Optimisation*
BSOCK	*Bees Swarm Optimisation Clustering based K-means*
SAF	*Saut Avec Flip*
NAF	*Suivant Avec Flip*
KBSO	*K-means base Bees Swarm Optimisation*
BKM	*Bees swarm optimisation base K-Means*

Introduction Générale

Introduction Générale

La fouille de données est un ensemble de méthodes qui permet l'extraction de connaissances à partir d'une grande masse de données.

On distingue 05 taches de fouille de données tel que la classification supervisée, la régression, la prédiction…etc., parmi d'elles la classification non supervisée.

La classification non supervisée permet de regrouper des données homogènes dans un même cluster.

Il existe plusieurs façons à regrouper les données : hiérarchiques partielles, …etc., dans ce présent projet, on traite la classification de données non supervisée et plus particulièrement la classification partielle.

K-means est l'algorithme le plus populaire des algorithmes de classification non supervisée et utilisable dans plusieurs applications réels tel que : la segmentation d'images, la recherche d'information….etc.

Dans ces derniers décennies, plusieurs versions de K-means ont été développé il y'à même des hybridations entre méta heuristique en citant : ISODATA, dynamique clustering algorithme…etc. Cependant, il a des problèmes, et l'un de ces problèmes l'initialisation aléatoire.

Dans un premier temps, on va faire une analyse de ces algorithmes après on proposera un algorithme basée sur un méta heuristique. A la fin et pour évaluer l'efficacité et la performance de l'algorithme proposé, on va faire une comparaison avec les algorithmes existants sur un ensemble de données.

Pour cela, ce présent projet est organisé comme suit :

Chapitre 1 : Etat de l'art de la fouille de données.

Chapitre 2 : Etat de l'art sur les méthodes d'optimisation et les métas heuristiques.

Chapitre 3 : Etudes détaillé sur les algorithmes de la classification non supervisée partielle.

Chapitre 4 : Conception d'un algorithme intelligent pour la classification non supervisée Partielle.

Chapitre 5 : Implémentation et expérimentation de l'algorithme proposé.

Chapitre 1 :

La fouille de données

«Les gens voient les choses comme
elles sont, et disent pourquoi ?
Mais, je les vois comme elles doivent
être. Et je dis pourquoi pas.»

J.F.K

1. Introduction

La convergence entre l'information et la communication produit une société assez riche de données et d'informations, ainsi le développement des supports et des techniques de stockage permettent de créer des entrepôts de données.

Grace à l'exploitation de données et l'apparition des entrepôts de données, l'extraction de la connaissance est devenue nécessaire dans notre ère.
De nombreuses méthodes sont apparues, permettent d'extraire des connaissances à partir de grandes base de données. Ces méthodes ont des origines diverses et souvent multiples. Certaines sont issues de statistiques d'autres proviennent de l'analyse de données.

L'intelligence artificielle est de plus en plus évoluée, plusieurs techniques ont été développées (des techniques d'apprentissage et de la résolution des problèmes), ces techniques ont donné une nouvelle vision d'extraction de connaissances, ainsi est née ce qu'on appelle la fouille de données.

La fouille de données est un ensemble de méthodes et des techniques destinées à l'analyse et l'exploitation des données, de façon automatique ou semi-automatique, afin de découvrir des règles, des relations, des structures particulières à travers une grande quantité de données.

On utilise la fouille de données dans plusieurs domaines : par exemple dans le marketing (pour le système de créations des profils et ciblage des clients potentiels et nouveau marché), dans le domaine de finance (minimisation de risque financiers), dans le bioinformatique (Analyse de génome et mise en point de médicaments) et dans le web (e-commerce et détection d'intrusion…).

Actuellement, plusieurs logiciels de fouille de données sont disponibles selon leur intérêt : SPAD, SAS, Enterprise Miner, SPSS, Weka…..etc. En fait, la fouille de données est l'une des étapes du processus d'extraction de connaissances à partir de données appelé KDD (Knowledge Discovery in Data bases), qui désigne tout le cycle de découverte de connaissances, Il regroupe donc la conception et les accès des grandes bases de données, et tous les traitements à effectuer pour extraire des connaissances de ces données.

Dans ce chapitre, on va expliquer le processus KDD ainsi quelques taches de fouille de données.

2. Le processus de découverte de connaissances KDD

2.1. Définition

Est un processus d'extraction de connaissances à partir de données **[1]**, il peut interpréter un grand ensemble de données brutes afin d'extraire des connaissances exploitables et a deux parties :

a. **Préparation des données :** Contient les étapes suivantes :

1. ***Traitement de données*** **:** Elle sert à collecter des informations et à organiser de ces informations dans une base de données.
2. ***Sélection de données*****:** Dans cette étape on sectionne les enregistrements et les attributs nécessaires pour le problème traité. Cette sélection permet de réduire l'espace de résolution.
3. ***Nettoyage de données*** **:** Elle sert de traiter et corriger :

➢ ***Les valeurs manquantes :*** Ce sont des champs qui n'ont aucune valeur par exemple

Si on a ce genre d'erreur on a 3 possibilités : **[2]**

- Exclure les enregistrements incompatibles: éliminer tous les enregistrements dans une valeur manquante.
- Remplacement : remplacer la valeur manquante par une valeur déterminée, soit par la connaissance que l'on a des données, soit par une source externe. le plus simple est de remplacer la valeur manquante par la valeur la plus fréquente (variables qualitatives) ou la moyenne ou la médiane (variables numériques). **[3]**
- Héritage : calculé si c'est possible.

➢ ***Les valeurs aberrantes :*** Ce sont des données incompatibles avec leur type. La méthode la plus pratique consiste à définir un espace compris entre la moyenne et l'écart-type puis à éliminer toutes les valeurs qui se trouvent à l'extérieur de l'intervalle [moyenne- écart, moyenne + écart-type].

Note :

$$\text{Moyenne} = \frac{\sum_{i=1}^{n} Xi}{n} \qquad \text{écart-type} = \sqrt{\frac{1}{n}(X^2 - \bar{X})^2}$$

Exemple:

ID	Nom	Prénom	Age
001	A	Y	15
002	D	F	13
001	A	Y	15
004	G	B	32
005	T	C	50
006	V	E	85
007	S	T	160
008	X	Q	?

Le nettoyage de données se fait comme suit :

- ✓ La valeur doublant : d'après le tableau l'enregistrement (001, A, Y, 15) est doublé, donc on laisse un et on supprime l'autre.
- ✓ La valeur aberrante : d'après le tableau l'âge=160 est une valeur aberrante, on calcule la moyenne et l'écart type. Ensuite on la remplace par une valeur compris entre [moyenne - écart, moyenne + écart-type].
- ✓ La valeur manquante : il y'a 3 possibilité :
  - ➢ Soit on élimine la ligne « 008 ».
  - ➢ Soit on la remplace par la moyenne.

3. ***Transformation et intégration de données*** **:** On applique des techniques sur les données et on les transforme sous des formes adaptées en problème donné.

b. Extraction des connaissances : Elle peut être décomposée comme suit :

1. ***Fouille de données :*** La fouille de données est le cœur du processus KDD **[4]** car elle permet d'extraire de l'information des données. Néanmoins, c'est souvent une étape difficile à mettre en œuvre, couteuse et dont les résultats doivent être interprétés et relativisés.
2. ***Interprétation la connaissance :*** consiste à analyser les connaissances obtenue par le processus de fouille. Ainsi à vérifier ses validités.

3. ***Présentation de la connaissance:*** Dans cette partie on va présenter les résultats obtenus à l'utilisateur afin de l'exploiter d'une application donnée.

2.2 Le schéma de KDD

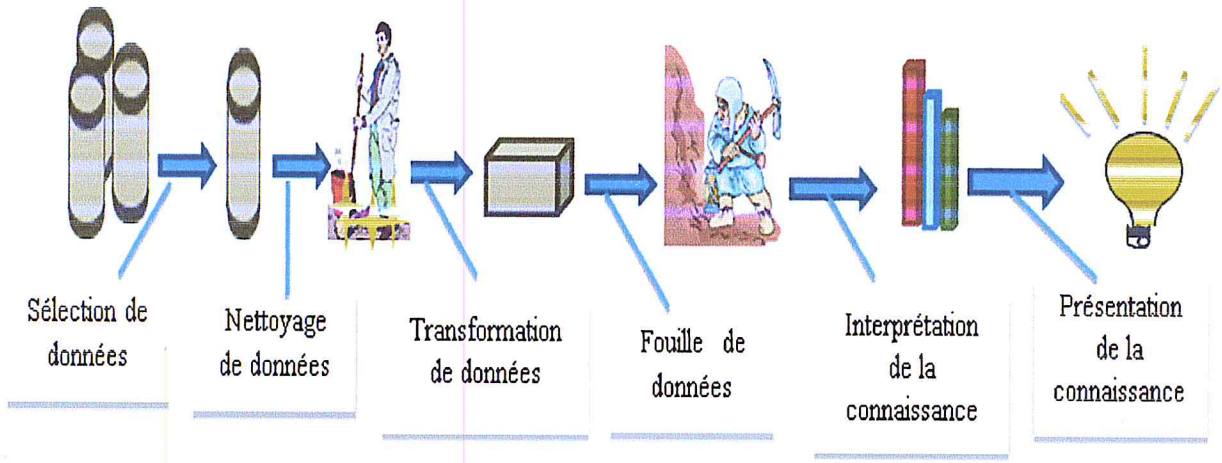


Figure 1. 1: Les étapes de processus KDD.

3. Définitions et types de données [3]

3.1. Définitions

Dans la fouille de donnée on utilise les différents genres de **donnée**.

Une donnée : C'est une valeur d'une **variable** pour un objet, elle se représente sous forme d'**attribut** et **exemple**.

Une variable : On appelle variable toute caractéristique d'une entité (personne, organisation, objet, événement…) qui peut être exprimée par une valeur numérique (mesure) ou codée (attribut).

Un attribut : Est un descripteur qui permet de représenter une caractéristique d'un exemple, il peut être de nature **qualitative** ou **quantitative [5].**

a) ***Un attribut qualitatif***: si on ne peut pas faire une moyenne (une couleur, une marque de voiture, ...).

b) ***Un attribut quantitatif*** : un entier, un réel, ... il peut représenter un salaire, une surface, un nombre d'habitants…

Exemple : Est un ensemble d'attribut formant un individu, et l'ensemble d'individu est la population à fouiller.

Exemple : Les transports à grande échelles :

La population est l'ensemble de villes.

Un individu est une ville.

Le nom est un attribut caractérisé par une ville.

3.2. Types de données [3]

Les données sont représentées sous forme d'attributs et d'exemples, un attribut est un descripteur qui permet de représenter une caractéristique d'un exemple, par contre un exemple est un ensemble d'attributs formant un individu, et l'ensemble d'individus est la population à fouiller.

Les données à fouiller peuvent être représentées selon différents types :

- Numérique continue : une variable peut prendre une valeur dans R

Exemple : la paye d'un employé X est 40000 DA.

- Numérique discrète : la valeur appartient à Z ou N

Exemple : l'âge d'un employé X est 32 ans.

- Catégorie : exemple {rouge, vert, blanc}.

- Binaire : la valeur peut prendre 0 ou 1

Exemple : 0 signifie que l'employé est une femme, 1 s'il est un homme.

- Chaine de caractère : par exemple : la spécialité d'un employé est « génie mécanique».

- Chaine de caractère : par exemple : la spécialité d'un employé est « génie mécanique».
- Arbre : par exemple : page HTML.

4. Similarité entre les données [7, 8]

4.1 Définition de distance

C'est une fonction qui précise la similarité des objets que l'on veut regrouper. Cette mesure est appelée distance ou métrique. Une distance est une formule qui prend deux points en entré et calcule un nombre positif reflétant la proximité éloignement de ces deux points.

4.2 Propriétés d'une distance

Chaque distance doit vérifier ces quatre points :

- ✓ **Symétrie : D(x,y)=D(y,x).**
- ✓ **Réflexivité : D(x,x)=0.**
- ✓ **Positivité : D(x,y) ≥ 0. Si x ≠ y.**
- ✓ **Inégalité triangulaire : D(x,y) ≤ D(x,z) + D(z,y).**

4.3 Les Mesure Usuelles de Similarité

Il existe plusieurs manières de calcule de similarité entre les données. On peut les classer selon le type de données utilisées comme suit :

4.3.1 ***Cas d'attributs numériques :*** On peut utiliser deux distances ;

- ***Distance euclidienne*** : $D(X_i, X_j) = \sqrt{\sum_{r=1}^{n} |X_{ir} - X_{jr}|^2}$
- ***Distance de Manhattan :*** $D(X_i, X_j) = \sum_{k=1}^{n} |X_{ik} - X_{jk}|$.

Exemple : A (2, 5), B (4,1). La distance euclidienne entre A et B est :

$D(A, B) = \sqrt{2^2 + 4^2} = \sqrt{20}$.

5. Les tâches de fouille de données [3,6]

Les Taches de fouille de données sont :

5.1. Classification

C'est un processus qui permet d'organiser un ensemble de données en classes, on distingue deux types de classification supervisée et non supervisée telles que :

a) La classification supervisée

C'est une opération qui permet de placer chaque individu de la population dans une classe parmi l'ensemble de classes déjà établies en fonction des caractéristiques de l'individu, parmi les techniques les plus utilisées : K plus proches voisins, réseau de neurones, arbre de décision.

Exemple :

- On dispose de donnée sur des patients atteint d'un cancer et d'autres sains.
- On déduit de leurs caractéristiques un modèle.
- Ce modèle est utilisé pour déterminer le risque pour d'autres patients de développer un cancer.

b) La classification non supervisée

Consiste à former des groupes à l'intérieur d'une population, contrairement à la classification, les groupes ne sont pas établies, parmi les techniques utilisées : K-means, CHA, Fuzzy-Cmeans….etc.

Exemple :

On dispose des données sur les clients (âge, nombre d'enfants, revenue, nombre d'achats, etc.). On regroupe les clients les plus proches en clusters. Ensuite, pour chaque cluster, on définit une offre commerciale destinée aux clients de ce cluster.

5.2. Estimation

Contrairement à la classe classification, le résultat d'une estimation peuvent d'obtenir une variable continue au lieu de discrète.

Exemple: Estimer salaire selon sexe, l'âge et l'ancienneté.

5.3. Prédiction

Elle ressemble à la classification et l'estimation, mais dans une échelle temporelle différente. Tous comme les taches précédentes, elle s'appuie sur le passé et le présent mais son résultat se situe dans un futur généralement précisé. La seule méthode pour mesurer la qualité de la prédiction est d'attendre !

Exemple: Prédire le gagnant de championnat de tennis par rapport de comparaison les résultats des joueurs.

5.4. Règles d'association

Consiste à trouver des relations relies entre les attributs.

Exemple: recherche des règles d'associations entre les pages web pour trouver les pages visitées et ce qui ne sont jamais visitées.

La Figure 1.2 représente les taches de fouille de données.

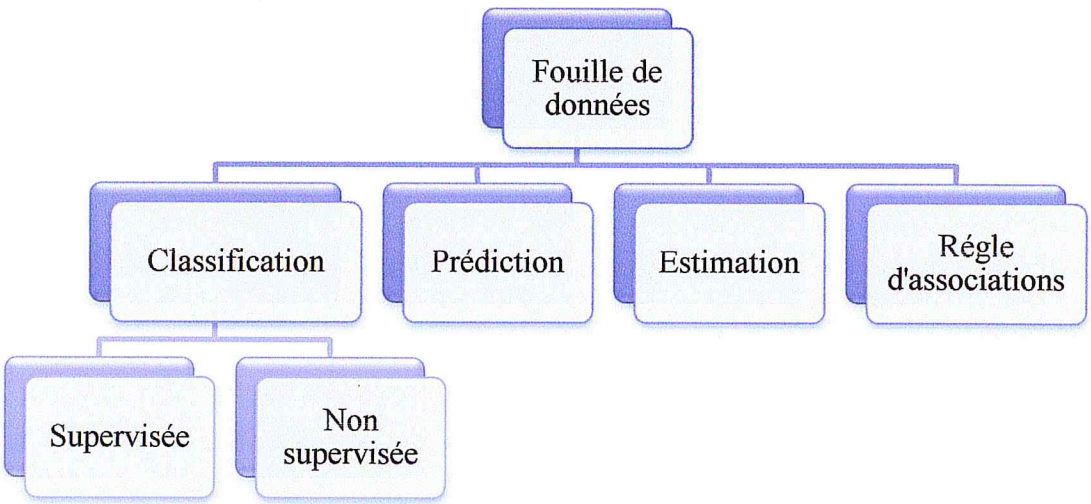


Figure 1. 2 : Les Tâches de la fouille de données

6. Les techniques de fouille de données

Les taches de fouille de données possèdent plusieurs techniques selon le tableau suivant :

La tâche de fouille de données	Les techniques utilisées
Classification supervisée	K plus proches voisins, les arbres de décision, les réseaux de neurones…
Classification non supervisée (clustering)	K-means, CHA…
Estimation	Réseaux de neurones…
Prédiction	Les arbres de décision, réseaux de neurones
Règle d'association	A priori…

Table 1. 1 : les techniques des taches de fouille de données.

6.1. Technique de classification non supervisée

a- Principe de l'algorithme K-means

L'algorithme K-means est largement utilisée en raison de sa théorie fiable [7], c'est un simple algorithme, il peut traiter efficacement les grands ensembles de données. Dans cet algorithme, chaque classe est représentée par la moyenne ou la moyenne pondérée qui est nommée le centroide.

K-means est un algorithme itératif. Il commence avec un ensemble de K points références choisis par l'utilisateur. Au début, les points sont partitionnes dans K classes : un point appartient à une classe si le point de référence de cette classe est le plus proche de lui. La mise à jour des points de référence et l'affectation des points de données aux classes sont réalisées pendant les itérations successives.

b- Description de l'algorithme K-means

Entrée: un échantillon de m enregistrements $X_1, \ldots, X_m$.

1- Choisir k centres initiaux $C_1, \ldots, C_k$
2- Répartir chacun des m enregistrements dans le groupe i dont le centre C_i est le plus proche.
3- Si aucun élément ne change de groupe alors arrêt et sortir les groupes
4- Calculer les nouveaux centres : pour tout i, C_i est la moyenne des éléments du groupe i.
5- Aller en 2.

Voici un schéma représentant les différentes phases de K-means.

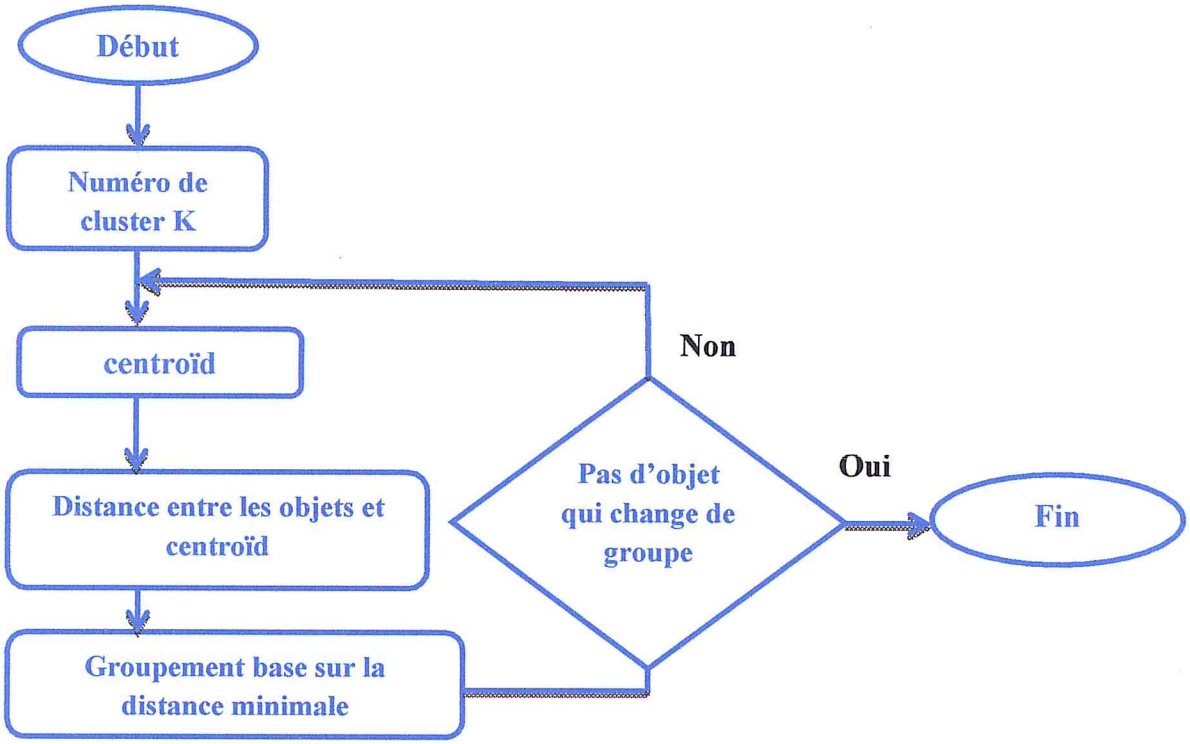


Figure 1. 3: Organigramme de l'algorithme K-means.

c- Calcule de complexité

Soit n est le nombre d'individus dans une population et k le nombre de clusters. La complexité de l'algorithme K-means est O (n^k log(n)) par ce qu'au pire des cas chaque individu visite tous les clusters, c'est comme si on fait toutes les combinaisons possibles de tous les clusters avec tous les individus.

d- Illustration

Exemple Théorique :

Soit l'ensemble de données qui est constituées de 8 points A, B,…., H de l'espace euclidien tels que : A(1,3) ,B(2,2) ,C(2,3) ,D(2,4) ,E(4,2) ,F(5,2) ,G(6,2) ,H(7,3)

→ Choisit K=2, c'est à dire on cherche à constituer deux groupes dans cet ensemble de 8 point.

→ On tire aléatoirement les 2 centres initiaux B et D sont choisis.

→ On repartit les points dans les deux groupes (nomes B et D) en fonction de leur proximité aux centres B et D.

→ Remarque : distance euclidien $(X_i, X_j)=\sqrt{\sum_{r=1}^{n}|Xir - Xjr|^2}$

→ On calcul les distances :

d (A,B)=$\sqrt{2}$=d(A,D)=$\sqrt{2}$=> on met A dans B ou D; soit B.

d(C,B)=1=d(C,D)=1 =>on met C dans B ou D; soit B.

d(E,B)=2<d(E,D)=$\sqrt{8}$=> on met E dans B.

d(F,B)=3< d(F,D)=$\sqrt{13}$=> on met F dans B.

d(G,B)=4< d(G,D)=$\sqrt{20}$=> on met G dans B.

d(H,B)=$\sqrt{26}$=d(H,D)=$\sqrt{26}$=> on met H dans B ou D; soit B.

Etape	**Etape 1**
	Centre D (2,4)
	Centre B (2, 2)
points	**Groupe**
A	B
B	B
C	B
D	D
E	B
F	B
G	B
H	B

→ On calcul les nouveaux centres pour l'étape 2, on obtient D et C1 (26/7,17/7) ou les coordonnées de C1 sont les moyennes des coordonnées des 7 autres points de B. le processus répète le même travail de la première itération jusqu'à trouver une stabilité.

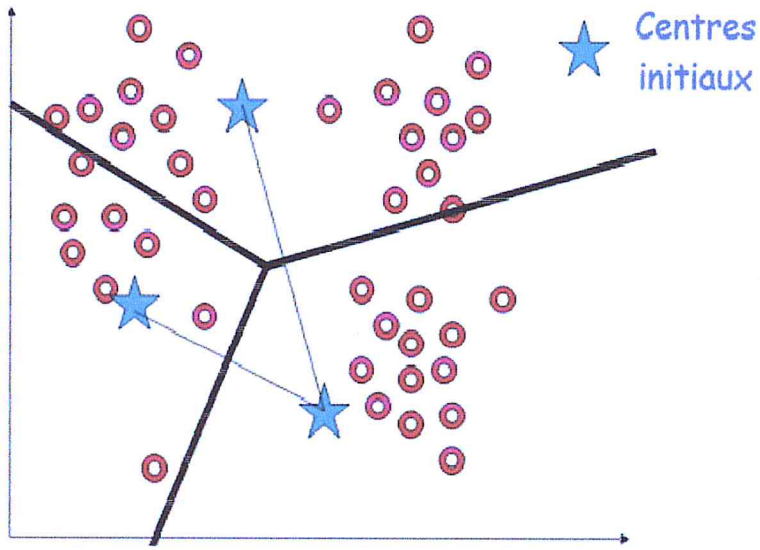
→ La table ci-dessous représente les résultats obtenus :

Etape	**Etape 1**	**Etape 2**	**Etape 3**	**Etape 4**
	Centre D (2,4)	**Centre D(2,4)**	**Centre C2 (5/3,10/3)**	**Centre C4 (7/4,3)**
	Centre B (2, 2)	**Centre C1 (26/7,17/7)**	**Centre C3 (24/5,11/5)**	**Centre C5 (22/4,9/4)**
points	**Groupe**	**Groupe**	**Groupe**	**Groupe**
A	B	D	C2	C4
B	B	C1	C2	C4
C	B	D	C2	C4
D	D	D	C2	C4
E	B	C1	C3	C5
F	B	C1	C3	C5
G	B	C1	C3	C5
H	B	C1	C3	C5

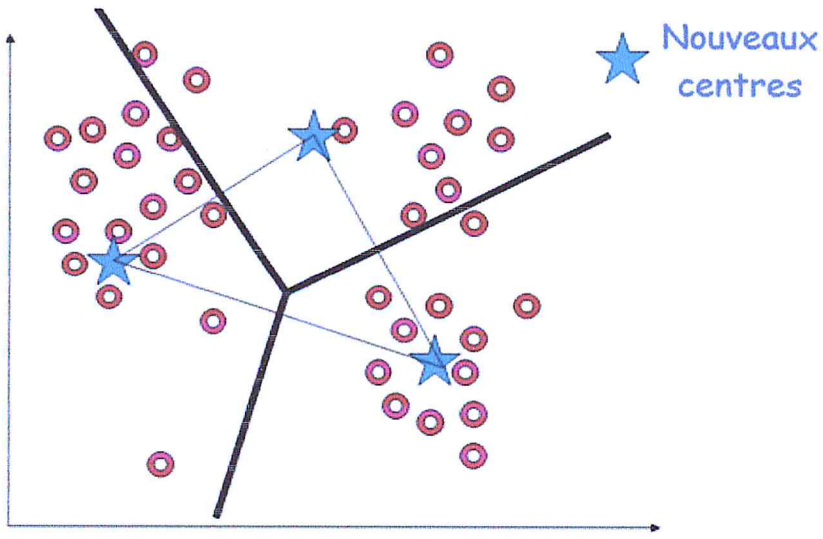
Exemple pratique [8] :

Voici un exemple explicatif de k-means sur un ensemble de données réelles. Les différentes étapes de processus sont présentées comme suit :

1erItération :



2emeItération :



Itération final :

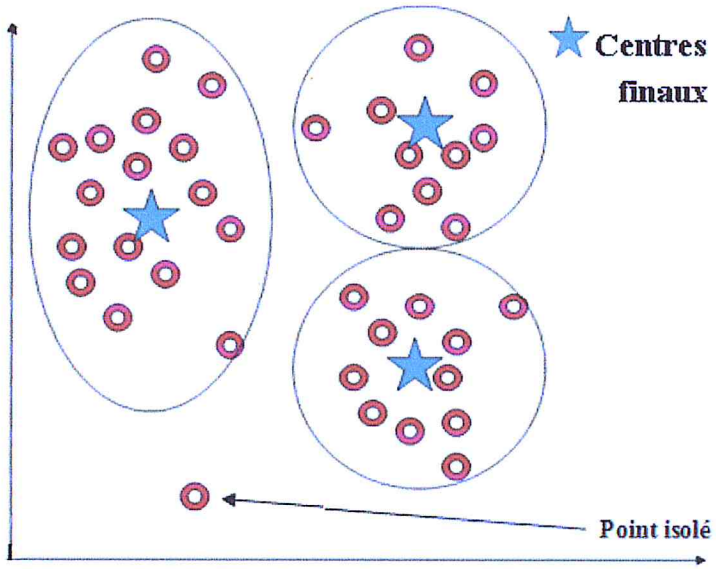


Figure 1. 4: Principe de k-means avec un dessin.

On voit bien à la fin de processus la convergence des données aux clusters adéquates.

e- Les avantages et les inconvénients de K-means [3, 8]

✓ ***Les avantages***

- ❖ Relativement extensible dans le traitement d'ensembles de taille importante.
- ❖ Facile à implémenter.
- ❖ Appliqué aux tous types de donnée.

✓ ***Les inconvénients***

- ❖ Applicable seulement dans le cas où la moyenne des objets est définie.
- ❖ Besoin de spécifier k, le nombre de clusters.
- ❖ Assez sensible aux points initialement sélectionnés, et il ne produit pas toujours le même résultat.

- Incapable de traiter les données bruitées.
- Les points isolés sont mal gérés.
- Peut produire des clusters vides.
- Très long.

6.2. Technique de classification supervisée

a- Principe de l'algorithme KPPV

C'est un algorithme qui permet la classification supervisée, son but est de déterminer la classe pour le nouvel individu en entrée, en utilisant les individus déjà classée.

b- Description de l'algorithme KPPV

Paramètre : le nouvel individu "X".

Un entier K entre 1 et le nombre d'individus déjà classes.

Début

Pour chaque individu "indiv" déjà classe.

Faire calculer la distance D(X, indiv).

Fin pour

Calculer K plus proche voisin(X).

Pour chaque indiv appartient à K plus proche

Faire calculer le nombre d'occurrence de chaque classe.

Fin pour

Attribuer à X la classe la plus fréquente ;

Fin

c- Calcule de complexité

Soit n est le nombre d'individus déjà classés. La complexité de l'algorithme kppv est O(n) car on fait n itérations dans la première boucle et k itérations dans la deuxième boucle tel que k<=n.

d- illustration

Soit le tableau suivant :

Individu	Age	Grade	classe
E1	24	5	Oui
E2	22	5	Oui
E3	21	5	Non
E4	21	4	Oui
E5	27	4	Non

→ En utilisant K plus proche voisins avec la distance Manhattan, on veut savoir la classe de l'individu "X"tel que : son âge est 22 et son salaire est 4.

→ Pour K=3 :

→ On calcule les distances entre l'individu" X" avec tous les individus déjà classées comme suit :

D(X, E1)=|24-22|+|5-4|=3, D(X, E2)=1, D(X, E3)=2, D(X, E4)=1,

D(X, E5)=5 ;

→ Après on sélectionne les 3 plus proches voisins de l'individu"X", soient E2, E3, E4.

→ Classe (E2)=classe (E4)= Oui alors nb d'occurrence de la classe Oui est " 2".

→ Classe(E3)=Non alors nb d'occurrence de la classe est "1".

✓ Donc la classe de l'individu "X" est "Oui".

e- Les avantages de KPPV

- La lisibilité des résultats, car on peut expliquer le choix en montrant les plus proches.
- Applicable à tout type de données.
- Peut traiter un grand nombre d'attributs.

7. Conclusion

En bref, la fouille de donnée est l'art d'extraire des connaissances à partir d'une grande masse de données.

L'avènement de la fouille de donnée a révolutionné le domaine du traitement des données en donnant un outil puissant qui permet à l'entreprise l'exploitation de l'information afin d'accélérer la prise de décision.

Grâce à la souplesse et la clarté de ses méthodes, la fouille de donnée est considérée comme le leader de la manipulation et le traitement de la donnée, elle fournit des résultats acceptables et compréhensibles.

La classification non supervisée est la tâche la plus riche en connaissances. Ainsi, l'algorithme k-means c'est l'algorithme le plus utilisé par sa simplicité et sa forte théorie. Cependant, en augmentant la taille de données, cet algorithme donne des résultats insatisfaisants. Pour cela les méthodes d'optimisation combinatoires telle que les métaheuristiques sont devenues une source indispensable pour la résolution de telle tâche. Dans Le chapitre suivant, on explique en détail quelques métaheuristiques les plus cités dans la littérature.

Chapitre 2:

Les métaheuristiques

«Ce n'est pas dans la science qu'est
le bonheur, mais dans l'acquisition
de la science.»

Edgar Allan Poe

1. Introduction

Les métaheuristiques forment un ensemble de méthodes utilisées en recherche opérationnelle pour résoudre des problèmes d'optimisation combinatoires, qui consiste de trouver l'optimum d'une fonction objective parmi un nombre dénombrable de solutions généralement grand.

On peut diviser les métaheuristiques en deux catégories : les méthodes de solution unique qui favorisent l'intensification à la diversification. D'autres méthodes sont basées sur population qui prend en considération la notion de diversification.

Ce chapitre montre les différentes métaheuristiques existant dans la littérature. Dans un premier temps nous nous pencherons leurs principes de fonctionnements généraux. Enfin, nous nous passerons en revue les métaheuristiques les plus usuelles, en analysant leur mode de fonctionnement, en évoquant les évolutions auxquelles elles ont donné lieu, et en soulignant leurs avantages et leurs inconvénients respectifs. Nous verrons à cette occasion toute la richesse et la variété des procédés mis en jeu dans les métaheuristiques.

2. Fonctionnement général des métaheuristiques[9]

Une métaheuristiqueest une stratégie (méthode) générale qui fournit des solutions de bonne qualité en temps raisonnable à des problèmes difficiles.

L'exécution des métaheuristiques se déroule principalement en quatre phases :

- Codage
- Intensification.
- Diversification.
- Mémoire.

2.1. Le Codage

Il y a principalement deux types de codage:

- ***le codage binaire***: C'est le plus utilisé. Chaque élément de la solution dispose du même alphabet binaire {0, 1}. Un élément est alors représenté par un entier long généralement de 32 bits.

Exemple : Dans le problème du voyageur de commerce, on peut coder les villes $c_i=\{c_1,c_2,c_3,\dots,c_n\}$ tel que i est la valeur de la ville. Les villes c_i sont donc une suite de bits (formée de zéro et un).

- ***le codage réel*:** On transforme la valeur de chaque élément de la solution à un nombre réel.

Exemple : Soit une solution composée de trois attributs binaires ($attribut_1$, $attribut_2$, $attribut_3$). Dans ce type de codage, chaque attribut « i », on le transforme à x_i tel que $x_i \in R$. La Figure (2.1) représente une illustration schématique du codage des variables réelles.

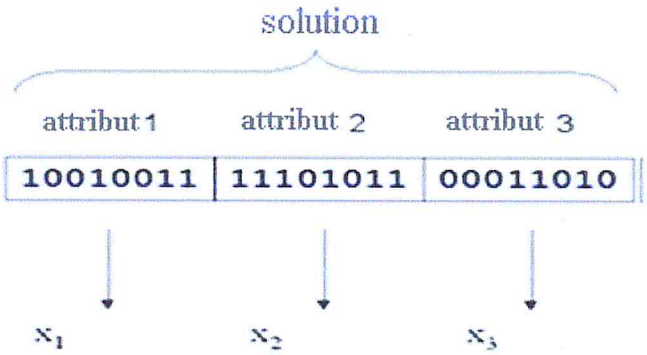


Figure 2. 1: Illustration schématique du codage des variables réelles.

2.2. ***L'intensification :*** C'est l'exploitation des solutions de la même région. D'autre terme, les solutions générées partagent plusieurs caractéristiques communes. Avec cette stratégie, on peut trouver rapidement l'optimum local d'une région spécifique. Cependant, l'optimum d'une région n'est pas forcément l'optimum global de l'espace considéré.

2.3. ***La Diversification :*** Contrairement à l'intensification, les solutions engendrées dans cette phase sont situées dans des différentes régions afin de bien exploiter l'espace de recherche. En effet, la diversification permet d'éviter de tomber à l'optimum local, Cependant, une hybridation entre les deux stratégies sera utile afin de converger à l'optimum globale.

2.4. ***La mémoire :*** Est le support de l'apprentissage permettant à l'algorithme de ne tenir compte que des zones où l'optimum est susceptible de se trouver et de garder en mémoire les résultats passés.

3. Métaheuristique à solution unique

Ce sont des méthodes itératives à solution unique basées sur un algorithme de recherche de voisinage qui commence avec une solution initiale, puis l'améliore pas à pas en choisissant une nouvelle solution dans son voisinage. **[10]**
Nous présenterons ici les méthodes les plus utilisées qui sont : recuit simulé et la recherche tabou.

3.1. Recuit simulé [10]

3.1.1. *Présentation*

Le recuit simulé est une technique d'optimisation de type Monte-Carlo généralisé à laquelle on introduit un paramètre de température qui sera ajusté pendant la recherche. Elle s'inspire des méthodes de simulation de Metropolis (année 50) en mécanique statistique.

La méthode du recuit simulé, appliquée aux problèmes d'optimisation, considère une solution initiale et recherche dans son voisinage une autre solution de façon aléatoire.

Au début de l'algorithme, un paramètre T, apparenté à la température, est déterminé et décroit tout au long de l'algorithme pour tendre vers 0. De la valeur de ce paramètre va dépendre la probabilité p d'acceptation des solutions détériorantes (plus la température 'T' est élevée, plus cette probabilité sera forte).

La performance du recuit simulé dépend, entre autres, de la règle de refroidissement (c'est-à-dire la décroissance du paramètre T) que l'on utilise. Un refroidissement trop rapide mènerait vers un optimum local pouvant être très mauvaise qualité. Un refroidissement trop lent serait très couteux en temps de calcul. Le réglage des différents paramètres (température initiale, nombre d'itération par palier de température, décroissance de la température,...) peut être long et difficile.

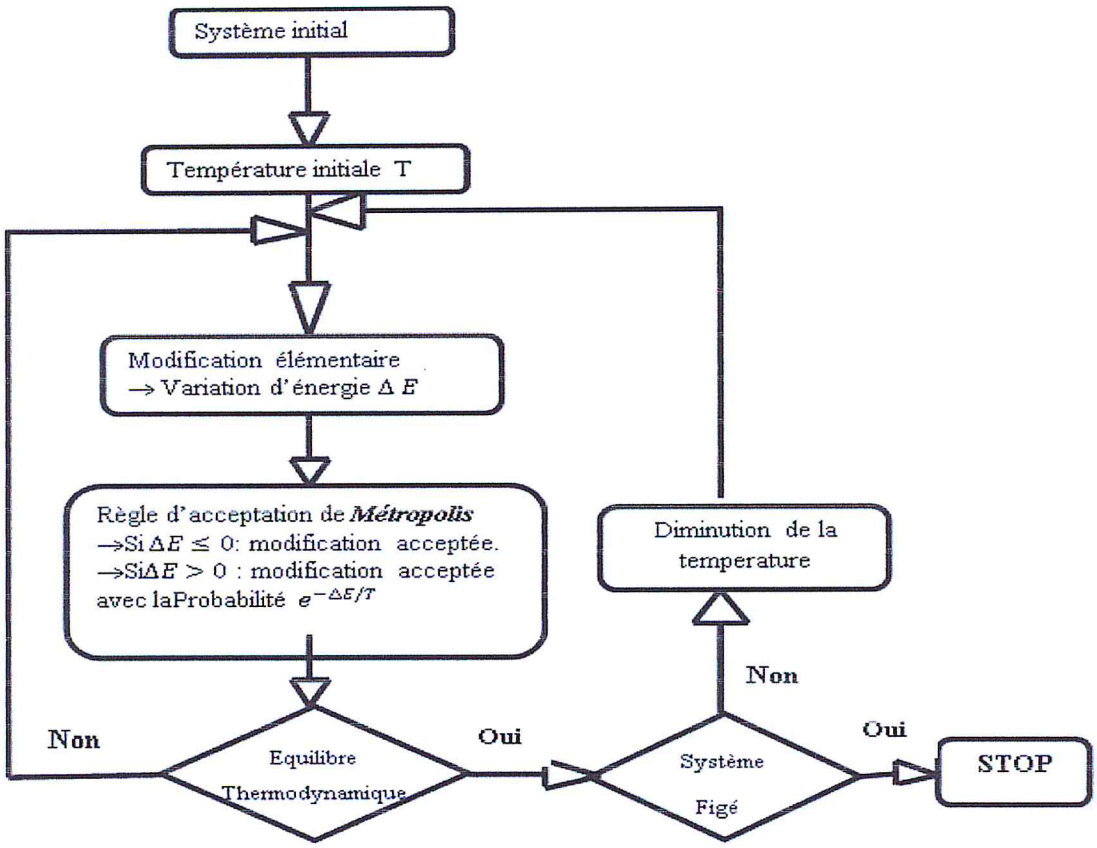


Figure 2. 2: Organigramme de la métaheuristique à recuit simulé. [9]

Les principales étapes de l'organigramme de la Figure (2.2) sont montrées ci-dessous :

→Température initiale : la température initiale est choisie arbitrairement par le programmeur. Tester l'algorithme pour différentes valeurs de température est souvent nécessaire.

→Modification élémentaire : la modification élémentaire permet de faire évoluer le système en abaissant légèrement sa température. Une fois cette modification effectuée, il faut calculer la nouvelle valeur de l'énergie du système et la différence avec ΔE.

→Règle d'acceptation de ***Métropolis*** : issue du domaine de la thermodynamique, cet algorithme permet d'y décrire l'évolution de tel système.

Dans notre cas si $\Delta E \leq 0$ alors on accepte la modification, sinon $(\Delta E > 0)$ on accepte la modification avec une probabilité $e^{-\Delta E/T}$.Ce second cas permet à l'algorithme de ne pas rester bloqué dans un minimum local lors du parcours de l'espace de recherche. Il permet en effet de garder des solutions qui paraissent mauvaises mais qui peuvent se révéler utiles par la suite.

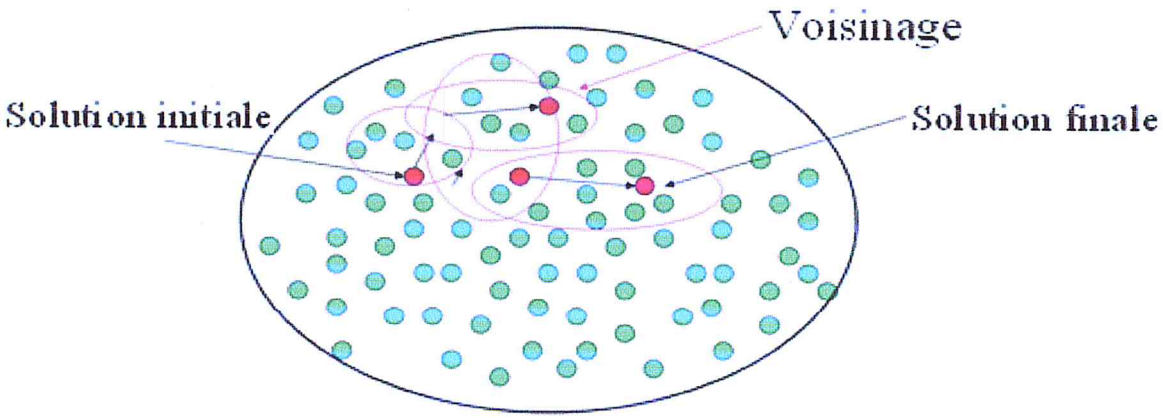


Figure 2. 3: Schéma d'évolution de l'algorithme de recuit simulé [14].

3.1.2. Avantages et inconvénients [9]

- ***Avantage*** **:** les avantages de recuit simulé sont résumés par :

  - ✓ Solutions de bonne qualité.
  - ✓ Méthode générale applicable à tout problème d'optimisation.
  - ✓ Très facile à programmer.
  - ✓ Ne nécessite pas de mémoire afin de trouver les espaces de recherche locaux suivants.

- ***Inconvénients*** **:**

  - ✓ Temps de calcul excessif dans certaines applications.
  - ✓ Nombre important de paramètres.
  - ✓ Il faut déterminer les paramètres à la main : température initiale, modification élémentaire….en testant divers valeurs.

3.2. Recherche avec tabou

3.2.1. *Présentation*

La méthode taboue a été développée par ***Glover*** (1986). Il s'agit de mettre chaque itération de la recherche locale dans une liste T dite tabou de manière temporelle (mémoire à court terme). Elle consiste à se déplacer de solution en solution en s'interdisant de revenir en une configuration déjà rencontrée.

La recherche tabou examine un échantillonnage de solutions de *N(s)* et retient la meilleure s' qui améliore la fonction objective. Habituellement on ne garde pas tous les déplacements en mémoire (trop couteux en ressources → mémoire à long terme) mais on va seulement empêcher l'accès à certaines solutions pendant un certain nombre d'itérations. Les tabous représentent les divers chemins parcourus par les différentes fonctions objectives. Ils sont réutilisés afin de diriger les explorations futures (diversification) vers des régions inconnues et sont stockés dans une liste *T* des *card (T)* dernières solutions trouvées. Cette liste est gérée de façon circulaire, comme une liste ***FIFO*** (**F**irst **I**n **F**irst **O**ut), on supprime le tabou le plus vieux pour ensuite insérer une nouvelle solution. Cette solution de stockage en mémoire permet alors de définir les tabous comme les "transformations"qui ont permis le passage d'une solution à une autre. **[13]**

L'organigramme général de la recherche tabou est expliqué dans la Figure (2.4)

Notation :

→ s_0 : solution initiale.

→ s^* : meilleure solution jusqu'à présent.

→ s' et s : nouvelle solution du voisinage de s^*.

→ $f(x)$: fonction objective à minimiser.

→ $f^* = f(s^*)$: valeur de la meilleure solution.

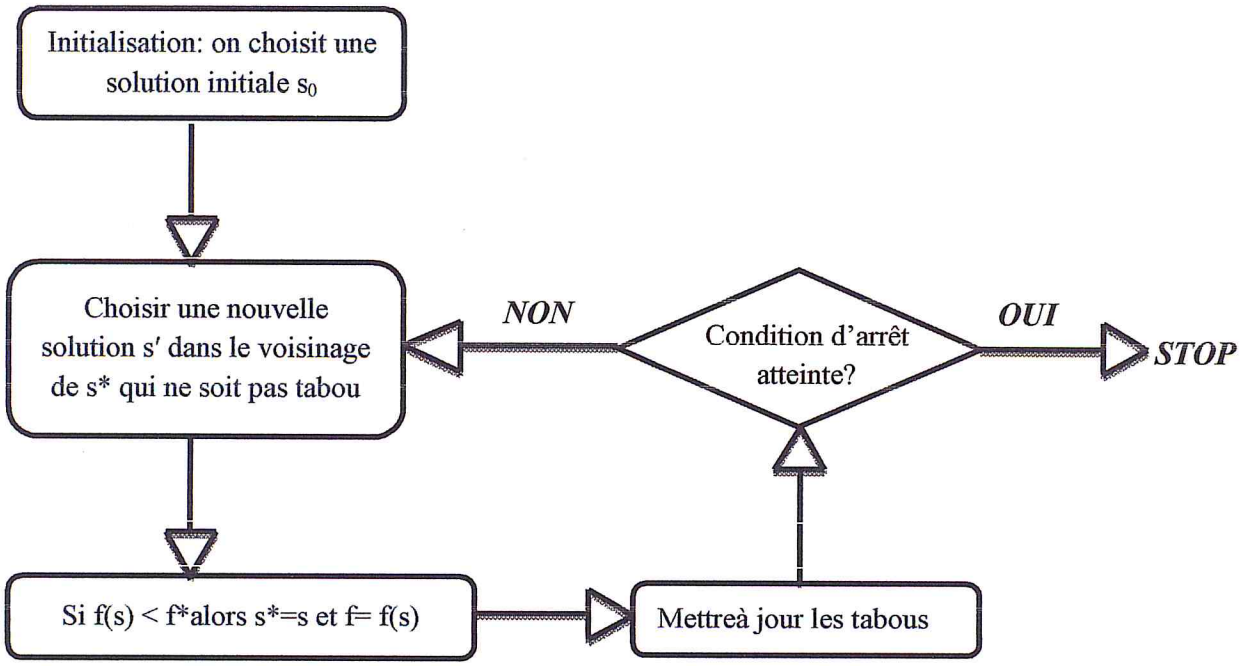


Figure 2. 4:Organigramme de la métaheuristique de recherche avec tabou. [9]

3.2.2. Avantages et inconvénients [9, 12]

- ***Avantage :***

  - ✓ Grande efficacité : elle a été testée avec succès sur les grands problèmes classiques (voyageur de commerce, ordonnancement d'ateliers) et elle est fréquemment appliquée sur les problèmes de constitution de planning, de routage, d'exploration géologique, etc.
  - ✓ Fonctionnement simple à comprendre.

- ***Inconvénients***

  - ✓ La méthode Tabou exige une gestion de la mémoire de plus en plus lourde. Afin de raffiner le procédé de la recherche tabou, on mise en place des stratégies de mémorisation complexe.
  - ✓ Paramètres peu intuitifs.
  - ✓ Aucune démonstration de la convergence.

4. Métaheuristique à solutions multiple

Les méthodes d'optimisation à population améliorent au fur et à mesure un ensemble de solutions d'une population à une autre. L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité. [10]

4.1. Les algorithmes génétiques [16]

Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle: croisements, mutations, sélection, etc. [18] cet algorithme a été inventé par John Holland à l'Université du Michigan dans le 1970, son premier livre est *"Adaptation in Natural and Artificial Systems"*, le principe de l'algorithme est de sélectionner les parents et crée ensuite les enfants. Les algorithmes génétiques peuvent être particulièrement utiles dans les domaines suivants : **[17]**

• Optimisation : optimisation de fonctions, planification, etc.
• Apprentissage : classification, prédiction, robotique, etc.
• Programmation automatique : programmes LISP, automates cellulaires, etc.
• Etude du vivant, du monde réel : marchés économiques, comportements sociaux, systèmes immunitaires, etc.

4.1.1. Principe de l'algorithme génétique

L'algorithme génétique repose sur une boucle qui enchaîne des étapes de *sélection* et des étapes de *croisement*. Dans un premier temps, à partir d'une population de α individus, on désigne ceux qui sont autorisés à se reproduire. On croise ensuite ces derniers, de façon à obtenir une population d'enfants, dont on peut faire muter aléatoirement certains gènes.

Figure 2. 5 : (a) Croisement [18], (b) Mutation [17].

La performance des enfants est évaluée, grâce à la fonction *fitness*, et l'on désigne, dans la population totale résultante parents + enfants, les individus autorisés à survivre, de telle manière que l'on puisse repartir d'une nouvelle population de α individus. La boucle est répétée, et l'on recommence une phase de sélection pour la reproduction, une phase de mutation, et ainsi de suite. Un critère d'arrêt permet de sortir de la boucle, par exemple un certain nombre d'itérations sans amélioration notable de la performance des individus. **[15]**

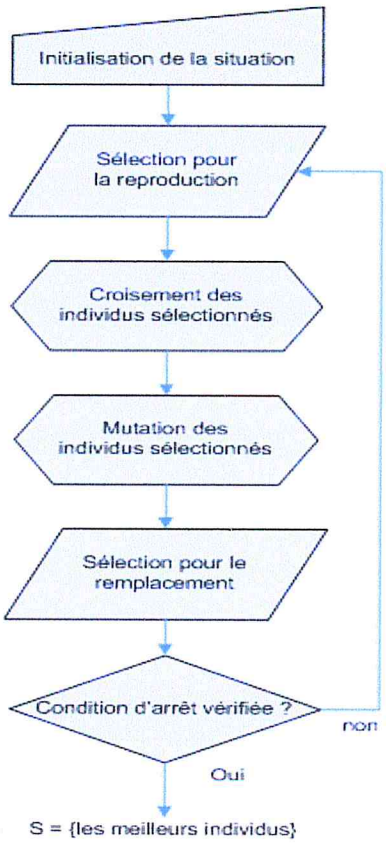


Figure 2. 6: Principe de fonctionnement d'un algorithme génétique. [15]

La Figure (2.7) explique clairement la reproduction des générations de l'algorithme génétique.

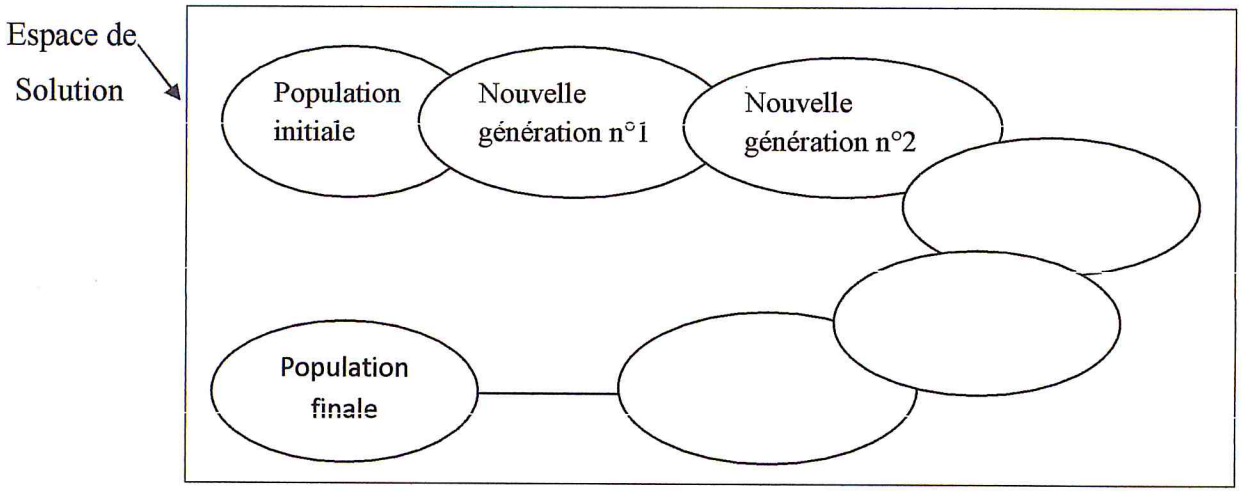


Figure 2. 7: Exploration de l'espace des solutions.

4.1.2. Les niveaux d'organisation d'un algorithme génétique [17]

Chaque paramètre d'une solution est assimilé à un gène. Un chromosome est une suite de gêne, on peut par exemple choisir de regrouper les paramètres similaires dans un même chromosome (chromosome à un seul brin) et chaque gène sera repérable par sa position.

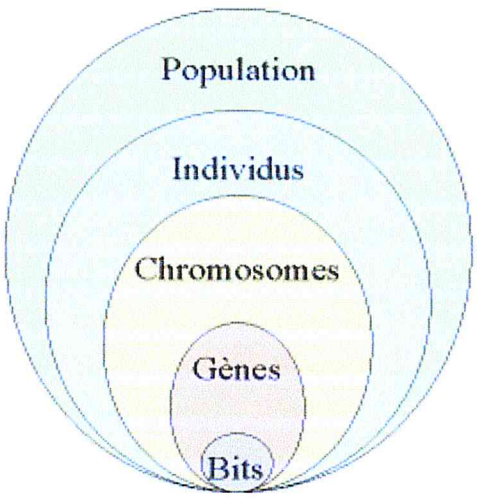


Figure 2. 8: Les cinq niveaux d'organisation d'un algorithme génétique. [17]

4.1.3. Avantages et inconvénients [15,20]

- *Avantage*

✓ Parallélisme intrinsèque

✓ Ils parviennent à trouver de bonnes solutions sur des problèmes très complexes, et trop éloignés des problèmes combinatoires classiques pour qu'on puisse tirer profit de certaines propriétés connues. Ils doivent simplement déterminer entre deux solutions quelle est la meilleure, afin d'opérer leurs sélections.

✓ Facilite d'implémentation.

✓ Beaucoup de littérature.

- *Inconvénients*

✓ les algorithmes génétiques sont coûteux en temps de calcul, puisqu'ils manipulent plusieurs solutions simultanément. C'est le calcul de la fonction de performance qui est le plus pénalisant, et on optimise généralement l'algorithme de façon à éviter d'évaluer trop souvent cette fonction.

✓ l'ajustement d'un algorithme génétique est délicat.

✓ Paramétrage important.

✓ Pas de stabilité des paramètres.

✓ Un autre problème surgit lorsque les différents individus se mettent à avoir des performances similaires : les bons éléments ne sont alors plus sélectionnés, et l'algorithme ne progresse plus.

4.2. Les algorithmes d'intelligence par essaim

Contrairement aux algorithmes génétiques qui ont été inspiré généralement par l'évolution d'une nouvelle population à partir d'une population donnée. L'intelligence par essaim et s'inspiré par le comportement de différentes espèces tel que les fourmis, les abeilles, les poissons, les oiseaux etc. Chaque individu de la population n'est pas intelligent mais la collectivité de ces espèces produit un certain degré d'intelligence. Ce qui suit, on va expliquer quelques métaheuristiques basées sur l'intelligence par essaim.[10]

4.2.1. Les colonies de fourmis [21]

Comme les algorithmes génétiques, les algorithmes de colonies de fourmi font évoluer une population d'agents. Cet algorithme est inspiré de la nature et de son organisation, il a été proposé par Marco Dorigo au début des années 90, cette technique offre un haut degré de flexibilité (la colonie s'adapte aux brusque de changement d'environnement) et robuste (la colonie continue à fonctionner lorsque certains individus échouent à accomplir leur tâche).

Son idée principale est d'imiter le comportement des fourmis réel pour résoudre un problème d'optimisation, lorsqu'une fourmi quitte son fourmilière (nid), elle se balade d'une manière aléatoire pour explorer son environnement à la recherche de nourriture, Chaque fourmi laisse en effet derrière elle une trace de phéromone(signal odorant) **[22]** pour lui indique son chemin de retour et guidé les autres fourmis à le plus court chemin de nourriture; les fourmis choisissant avec une plus grande probabilité les chemins contenant les plus fortes concentrations de phéromones, il se forme ainsi ces « autoroutes » à fourmis, naturellement des milliers de fourmis les font simultanément.

La Figure (2.9) explique le fonctionnement général de recherche de nourriture.

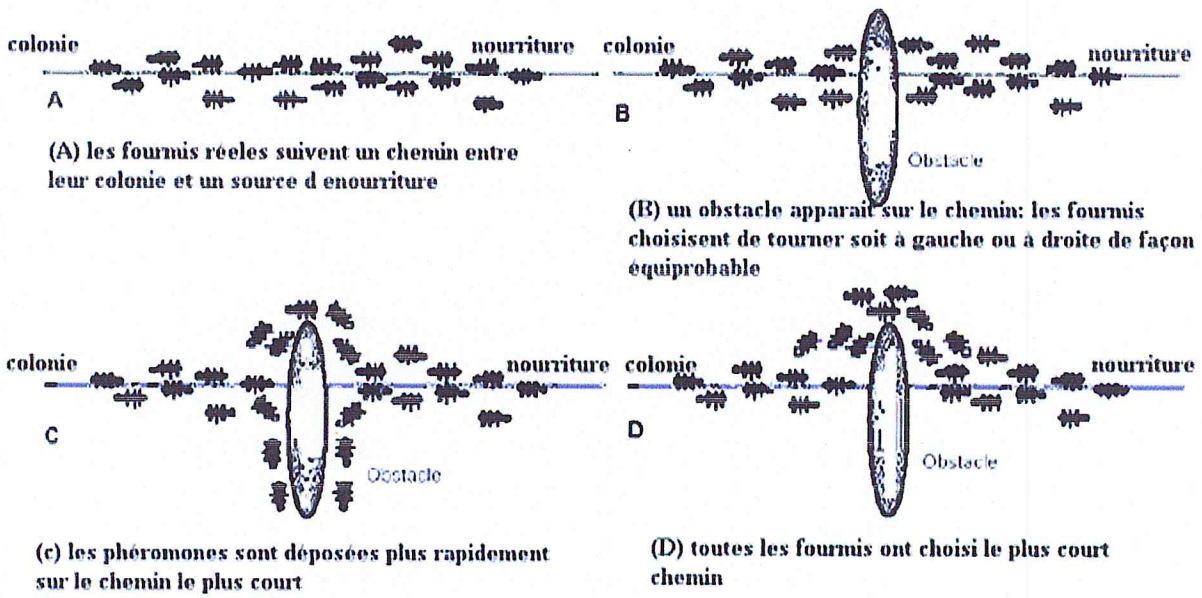


Figure 2. 9: La recherche du chemin optimal entre la colonie de fourmis et la nourriture [23].

4.2.2. Algorithme de colonie de fourmi [24]

L'objectif de cet algorithme est de trouver la tournée la plus courte pour un ensemble des villes données.

Pour t=1,…, t_ {max}

  Pour chaque fourmi k=1,…, m

  Choisir une ville au hasard v_h

    Pour chaque ville non visitée v_i

      Choisir une ville v_j, dans la liste des villes restantes, selon la règle

      Aléatoire de transition

    Fin pour

    Déposer une piste sur le trajet entre v_h et v_i

  Fin pour

  Évaporer les pistes

Fin pour

4.2.3. Description des éléments de l'algorithme

- Dans cet algorithme, à chaque itération t($1 \leq t \leq t_{max}$), chaque fourmi K (K=1,..., m) parcourt le graphe et construit un trajet complet de n=|C| tel que C représente l'ensemble de toutes les villes.
- On choisit une ville au hasard v_h.
- Parmi les villes non visitées v_i, La fourmi choisi la ville la plus proche v_j et éviter les villes trop lointaines.
- Apres un tour complet, chaque fourmi laisse une certaine quantité de phéromone sur l'ensemble de son parcours.
- Ensuite, le processus d'évaporation des pistes de phéromones, permet au système « d'oublier » les mauvaises solutions.
- La règle « aléatoire de transition » est une heuristique spécialisé dont le seul et unique but est d'obtenir de meilleures diversification des trajets. Cette heuristique peut être extrêmement simple par exemple, prendre la ville la plus proche.

4.2.4. Avantage et inconvénients [24]

- ***Avantage :***

  - ✓ Très grande adaptabilité.
  - ✓ Parfaite pour les problèmes représentés par des graphes.

- ***Inconvénients :***

  - ✓ Un état de blocage peut être arrivé.
  - ✓ Temps exécution parfois long.
  - ✓ Ne s'applique pas à tous type de problèmes.

4.3. Essaim d'abeille

4.3.1. Présentation [21]

La colonie des abeilles est un algorithme d'optimisation stochastique P-métaheuristique qui appartient à la catégorie « intelligence par essaim ». Dans les dernières décennies, plusieurs études basées sur différent comportements de colonie d'abeilles sont développées pour résoudre les problèmes complexes d'optimisation combinatoire.

L'algorithme de colonie d'abeilles est inspiré par le comportement des abeilles : le miel, car il expose plusieurs caractéristiques qui peuvent être utilisées comme un modèle d'intelligence collective.

Principalement les algorithmes des abeilles sont classés aux trois différents modèles :

- Recherche de nourriture.
- Recherche un site pour le nid.
- Mariage dans la colonie des abeilles.

Chaque modèle est définit pour donner des comportements à des tâches spécifiques.

4.3.2. Les abeilles dans la nature [21]

Les abeilles domestiques sont, comme les fourmis, des insectes sociaux elles ne vivent pas individuellement mais aux groupes. Ces insectes volant natals d'Europe, milieu de l'est et l'Afrique ont été introduit par l'apiculteur dans le reste du monde. Il y a plus que 20.000 espèces connues qui habitent dans des régions des fleurs et vivent dans des colonies sociales après avoir choisir leur nid et dans ce dernier il y a entre 60.000 et 80.000 habitants.

L'abeille est caractérisée par la production d'une complexe substance : le miel, et la construction de son nid à l'aide de la cire. L'abeille se nourrit de nectar qui est la source de son énergie dans sa vie, et utilise le pollen comme une source de protéine derrière leur ruche.

Généralement, la colonie des abeilles contient un reproducteur femelle s'appelle la reine, et des milliers de males et beaucoup de femelle stérile nommée ouvrières (travailleuses). Apres l'accouplement avec plusieurs males, la reine pond beaucoup de petites abeilles qui s'appellent larve. Parmi ces larves, la future reine qui est choisi par les ouvrières.

a) ***La reine :*** La reine est la seule femelle fertile de la communauté, c'est la plus grosse de toutes les abeilles. Elle est donc la mère de tous les individus qui la composent : les faux bourdons, les ouvrières et les futures reines. Sa capacité à pondre est très importante. Sa production journalière dépasse souvent 1500 œufs, dont la masse est égale à celle de son propre corps. La reine n'a pas non plus les appendices des ouvrières: paniers à pollen, glandes sécrétrices de cire et sac à miel bien développé. Sa nourriture quasi exclusive est une sécrétion, appelée gelée royale, produite par les glandes situées dans la tête des ouvrières. La durée de vie moyenne de la reine est de un à trois ans.

b) ***L'ouvrière :*** Les ouvrières sont toujours beaucoup plus nombreuses que les mâles. Dans la ruche d'une région tempérée, le nombre d'ouvrières est compris entre 8 000 et 15 000 individus au printemps, mais peut dépasser 80 000 au début de l'été. Les ouvrières sont incapables de s'accoupler et donc de se reproduire. Elles sécrètent la cire, construisent les alvéoles, récoltent le nectar, le pollen et l'eau, transforment le nectar en miel, nettoient la ruche et la défendent contre les prédateurs. Les ouvrières dont l'espérance de vie est d'environ six semaines. Pendant les trois premières semaines de leur vie adulte, les ouvrières s'emploient, entre autres, à construire les

rayons, nettoyer et polir les alvéoles, nourrir les jeunes et la reine, contrôler la température, évaporer l'eau du nectar jusqu'à ce qu'il prenne une consistance de miel épais. Après cette période, elles vont récolter le nectar sur les fleurs et défendent la ruche.

c) ***Le male :*** Le male ne travaille pas, est incapables de se nourrir lui-même et ne peuvent même pas défendre la colonie, puisqu'il n'a pas de dard, il présente entre 300 et 3000 dans la colonie. Sa seule fonction est de s'accoupler avec la reine. Quand elle quitte la ruche, les mâles sont attirés par son odeur. Après l'accouplement, le mâle meurt de faim rapidement. la durée moyenne de sa vie est 90 jours.

Le comportement des abeilles dans la colonie (la sélection de nid, la nourriture, la production du miel et le mariage…) sont des activités principales dans la vie des abeilles. Cela fait attirer les chercheurs pour créer des algorithmes d'optimisation basés sur le principe des abeilles réelles.

4.3.3. La sélection de la location du nid

Les sociétés d'abeilles vivent dans des nids appelés ruches. Dans le dernier du printemps ou au début d'été, la reine s'envole accompagnée des milliers d'ouvrières, pour aller construire une autre colonie. Ils se devisent en deux groupes : la reine avec la moitié des ouvrières et les filles de la reine avec le reste des ouvrières. Pour la sélection centaine des abeilles de scout explorent certaine site de nid, les autres abeilles restent inactives, probablement conservent l'énergie d'essaim jusqu'à prendre une décision, et migres pour la sélection du nouveau nid. Les abeilles responsables de recherche de nourriture indiquent les différents nids par plusieurs danses qui s'appellent « la danse remué ».

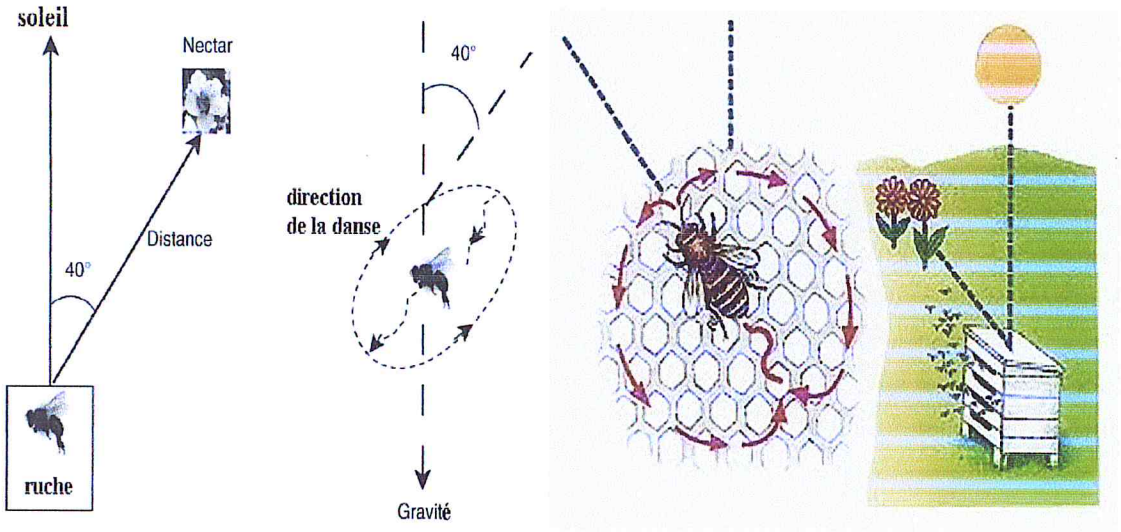


Figure 2. 10: La sélection de la location du nid des abeilles [21].

La direction est indiquée par un angle à partir du soleil, la distance est définie par la durée de remuement de la danse, il tourne 40◦ à droite indique la source de nourriture, et 40◦ à droite la direction du soleil (extérieur de la ruche). A travers la figure (2.10), l'abeille tourne verticalement à partir le centre, et forme un angle avec son corps par une vibration. Cet angle relie la ruche avec le nectar.

4.3.4. La nourriture

Le rôle principal de l'abeille qui cherche la nourriture est de trouver une source très riche de nourriture pour obtenir une quantité maximale de nectar. Pour cela il y a des types différents d'abeilles :

1. ***Abeille en chômage :*** Ces abeilles n'ont aucune connaissance de source de nourriture, et aucune vue de source de nourriture à exploiter. Donc ce type d'abeilles reste dans la ruche et attend la danse exercée par les autres abeilles pour établir la source de nourriture.
2. ***Abeille employé :*** Est associé avec une source de nourriture particulière. Elle a une connaissance suffisante pour connaitre la source de nourriture. Elle trouve et exploite

la source, mémorise la location et charge une partie de nectar et le décharge dans la ruche. Pour mentionner la qualité du nectar s'il est mauvais ou excellent.

4.3.5. Algorithme de recherche de nourriture

Initialisation aléatoire de toutes colonies des abeilles ;
Evalué la fitness de la population des abeilles ;

Répéter /*Formé une nouvelle population */

Sélectionner les sites de recherche ; Déterminer la taille de la pièce ;
Recruter les abeilles pour sélectionner des sites et évaluer leur fitness ;
Select l'abeille représentative de chaque pièce ;
Assigne le reste des abeilles a la recherche aléatoire et évaluer leur fitness ;

Jusqu'à critère d'arrête

Quand les butineuses repèrent des fleurs, elles retournent à la ruche et préviennent les autres butineuses. Pour leur indiquer où se trouve les fleurs, elles se mettent à « danser », utilisant un vrai langage du corps très codé.

→ Quand la source de nourriture est située à moins de 100 mètres de la ruche, les danseuses tournent sur elles-mêmes vers la gauche et vers la droite, en décrivant un cercle.

→ Si la nourriture est plus éloignée, elles décrivent un 8. L'abeille avance tout droit en agitant son abdomen, et décrit un demi-cercle de part et d'autre d'une ligne. Cette ligne indique la direction à suivre, et le nombre de tours donne la distance par rapport à la ruche.

Si la source de nourriture est très importante (par exemple si les butineuses ont trouvé un champ de fleurs), la danse est très rapide.

La figure ci-dessous (2.11) représente la danse d'une abeille.

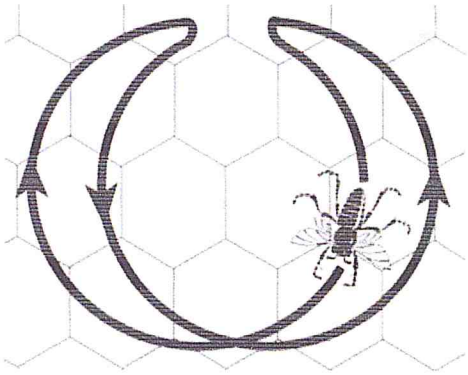


Figure 2. 11: La danse frétillante, appelée aussi en huit. [25]

4.3.6. Algorithme de Mariagedes Abeilles

Initialisation aléatoire de la reine;

Améliorer la reine par les ouvrières (S-métaheuristique) ;

Pour définir le nombre maximum du mating-flights

Faire

Initialise énergie and rapidité ;

Tan qu'énergie de la reine > 0

Faire

La reine se déplace et choisi les males ;

Si le male est sélectionné

Alors

Ajouter son sperme dans l'ovaire de la reine ;

Modifier l'énergie et la rapidité de la reine;

Fin tan que

Produire les larves avec le croisement et mutation ;

Utiliser les ouvrières (S-métaheuristique) pour l'occupation des larves ;

Modifier la fitness des ouvrières;

Si la meilleure larve adéquate la reine

Donc

Remplacer les chromosomes de la reine avec les chromosomes de la meilleure larve ;

Tuer tous les larves ;

Fin pour

4.3.7. Avantage et inconvénients

- ***Avantage :***

✓ Très efficace dans la recherche des solutions optimales.
✓ Surmonte le problème de l'optimum local.
✓ Facile à implémenter.

- ***Inconvénients :***

✓ L'utilisation de plusieurs paramètres réglables.
✓ Sensible à des problèmes extrêmement difficiles.
✓ L'optimisation de fonction.

5. Algorithmes à essaim de particules (PSO) [10]

Les algorithmes d'optimisation par essaim de particules (PSO) ont été introduits en 1995 par Kennedy et Eberhart comme une alternative aux algorithmes génétiques. Ces algorithmes sont inspirés des essaimes d'insectes (ou des bancs de poissons ou des nuées d'oiseaux) et de leurs mouvements coordonnés. En effet, tous comme ces animaux se déplacent en groupe pour trouver de la nourriture ou éviter les prédateurs, les algorithmes à essaim de particules recherchent de solutions pour un problème d'optimisation. Les individus de l'algorithme sont appelés particules et la population est appelée essaim.

Dans cet algorithme, une particule décide de son prochain mouvement en fonction de sa propre expérience, qui est dans ce cas la mémoire de la meilleure position qu'elle a rencontrée, et en fonction de son meilleur voisin. Ce voisinage peut être défini spatialement en prenant par exemple la distance euclidienne entre les positions de deux particules. Les nouvelles vitesses et direction de la particule seront définies en fonction de trois tendances : la propension à suivre son propre chemin, sa tendance à revenir vers sa meilleure position, et sa tendance à aller vers son meilleur voisin. Les algorithmes à essaim de particules peuvent s'appliquer aussi bien à des données discrètes qu'au des données continues.

6. Classification des métaheuristiques

On peut classer les métaheuristiques en deux grandes classes : les métaheuristiques à solution unique (évoluant avec une solution) et celles à des solutions multiples (évoluant avec plusieurs solutions).Une classification non exhaustive des méthodes métaheuristiques est illustrée dans la figure ci-dessous (Figure2.12).

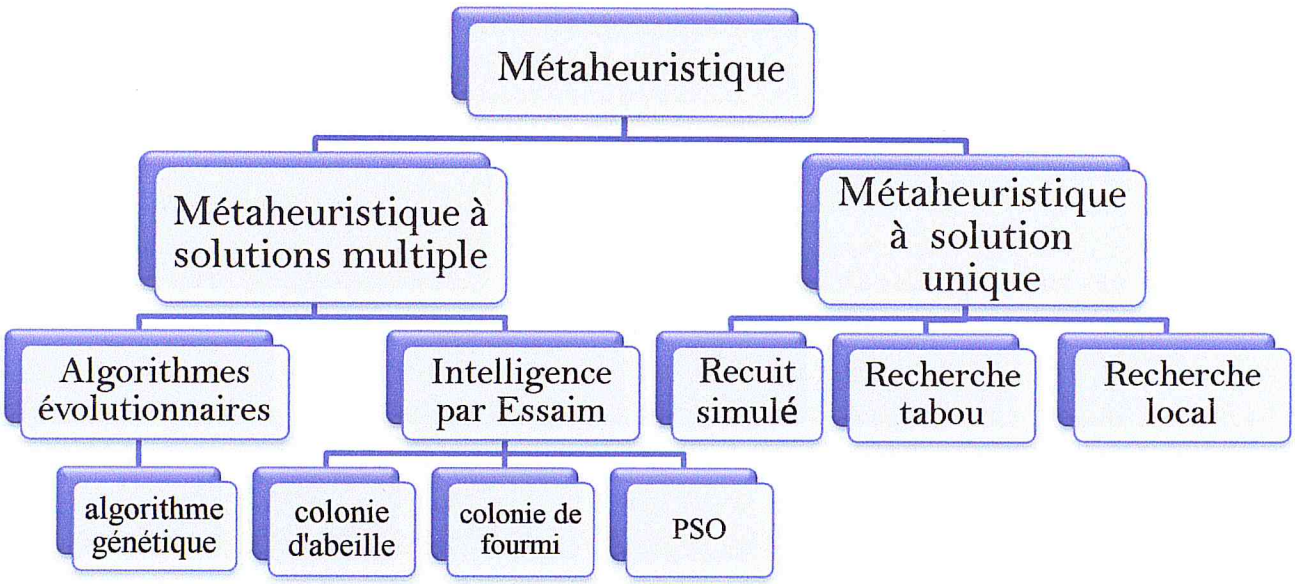


Figure 2. 12: Classification des métaheuristiques.

7. Conclusion

Les métaheuristiques constituent une classe de méthodes approchées adaptables à un grand nombre de problèmes d'optimisation combinatoire. Mais, si l'on a pu constater leur grande efficacité sur de nombreuses classes de problèmes, il existe en revanche très peu de résultats permettant de comprendre la raison de cette efficacité, et aucune méthode particulière ne peut garantir qu'une métaheuristique sera plus efficace qu'une autre sur n'importe quel problème.

Concrètement, certaines métaheuristiques présentent l'avantage d'être simples à mettre en œuvre, comme nous l'avons vu avec le recuit simulé ; d'autres sont plutôt bien adaptées à la résolution de certaines classes de problème, très contraints, comme le système de colonies de fourmis. La qualité des solutions trouvées par les métaheuristiques dépend de leur paramétrage (il faut éviter que les algorithmes ne convergent trop rapidement vers un optimum local), et de l'équilibre à trouver entre un balayage de tout l'espace des solutions (diversification) et une exploration locale poussée (l'intensification). Le choix d'une bonne représentation, d'un bon voisinage, sont également des facteurs influencent sur l'efficacité de la méthode choisie.

Depuis une dizaine d'années, l'hybridation des métaheuristiques est devenue un enjeu important, à tel point que toutes les différentes métaheuristiques que nous avons examinées, recuit simulé, Recherche Tabou, algorithmes génétiques et plus particulièrement les essaims d'abeilles, sont maintenant perçues comme des points de départ au développement de nouveaux algorithmes d'optimisation. Les différentes caractéristiques de ces métaheuristiques (mémoires, listes de candidats, populations d'agents, transformations de la fonction objectif, etc.) sont recomposées entre elles pour former de nouvelles méthodes, dont la liste s'allonge.

Chapitre 3 :

La classification non supervisée

« Tous corps qui bouge sera laisser en mouvement, tous corps qui sommeille sera mis au placard.»

Ross-Newton

1. Introduction

La classification non supervisée (clustering) est une tache de fouille de données qui permet de regrouper les objets les plus similaires dans le même cluster, et de mettre les objets dissimilaires dans des clusters différents.

Dans le premier chapitre, on a vu en détail le principe de l'algorithme K-means. Le problème de K-means réside à la difficulté du choix des centres de gravité initiales, dans le deuxième chapitre, on a expliqué quelques métaheuristiques les plus utilisées dans l'optimisation combinatoire.

Dans ce chapitre, on va voir un état de l'art des algorithmes d'hybridation entre des métaheuristiques et l'algorithme K-means. Cette hybridation nous permet de régler le problème d'initialisation des centres de gravité de K-means.

2. Les algorithmes de classification non supervisée basée à solution unique

2.1. L'algorithme "Clustering du données catégorielles utilisant la recherche tabou" [26]

2.1.1 Présentation

Le partitionnement d'un ensemble d'objets en groupes homogènes est une opération fondamentale en science des données. Tel que, cette méthode regroupe les données les plus similaires dans le même cluster. L'algorithme K-means est préférable pour l'opération de regroupement à cause de sa efficacité de regrouper les données. Mais le K-means travaille uniquement sur les valeurs numériques.

L'algorithme proposé utilise la recherche tabou basé sur l'algorithme de clustering K-means, tel que son principe est pour améliorer le K-means et l'utilisé dans les domaines catégoriques et les domaines avec les valeurs numérique et catégorique. Cet algorithme peut explorer l'espace de recherche à partir de minimum locale pour aller vers le minimum globale. Pour avoir trouvé le minimum global dans les problèmes d'optimisation combinatoire, les méthodes proposées de la recherche taboue sont utilisées pour guider le processus de recherche dans les régions difficiles, est parmi ces méthodes sont :

1. ***Configuration :*** Est une affectation des valeurs aux variables. Cela est une solution au problème d'optimisation.
2. ***Mouvement :*** Est une procédure spécifique pour obtenir une solution d'essai qui est faisable au problème d'optimisation, et qui est associés à la configuration courante.
3. ***Les voisins :*** Est l'ensemble de tous les voisins, qui sont les « Solutions adjacentes » qui peut être atteint à partir de n'importe quel configuration courant. Elle peut également inclure des voisins qui ne sont pas satisfaire aux conditions usuelles possibles donnés.
4. critères d'expiration sont des règles qui déterminent quand le tabou peut être remplacé, supprimé.

Les résultats les plus importants dans ce travailles sont les méthodes proposées, tel que ces méthodes permet l'efficacité de recherche tabou pour être utilisé dans le regroupement des données. Les résultats ont montré que la recherche taboue basée sur k-means est efficace pour récupérer les structures de regroupement nécessaire à partir des données catégoriques.

2.2. L'algorithme de recherche locale pour le problème de clustering [27]

2.2.1. Présentation

Le regroupement des données est considéré comme un problème d'optimisation combinatoire. Recherche locale offre une approche simple et efficace pour de nombreux autres problèmes d'optimisation combinatoire. Il est donc surprenant de voir combien rarement il a été appliqué au problème de clustering. Au lieu de cela, les meilleurs résultats de clustering ont été obtenus par des techniques plus complexes comme la recherche taboue et les algorithmes génétiques au coût élevé de l'exécution. Un nouvel algorithme de recherche locale proposé pour le problème de clustering. L'algorithme est facile à mettre en œuvre, suffisamment rapide et compétitive avec les meilleures méthodes de clustering. La facilité de mise en œuvre, il est possible d'adapter l'algorithme pour diverses applications de clustering avec différentes mesures de distance et de critères d'évaluation.

Le problème général de regroupement comprend trois sous-problèmes:

- La sélection de la fonction d'évaluation.
- La décision du nombre de groupes dans la grappe.
- Le choix de l'algorithme de classification.

Dans ce travail, le nombre de grappes (groupes) est fixé préalablement. La fonction d'évaluation dépend de l'application et le type d'objets de données. La décision est généralement introduites par le chercheur de la zone d'application, mais analytique il existe des méthodes pour aider à la décision.

La méthode utilisée pour générer un regroupement. C'est la plus citée et la plus utilisée est l'algorithme k-means. L'algorithme est facile à mettre en œuvre et il donne des résultats raisonnables dans la plupart cas. Malheureusement, l'algorithme ne fait que des changements locaux au regroupement original, et ça coince au premier minimum local. La qualité de la classification finale est donc très dépendante de l'initialisation.

De meilleurs résultats ont été obtenus par des méthodes d'optimisation telle que les algorithmes génétiques (GA) et la recherche Taboue (TS). La propriété commune de ces méthodes est qu'elles considèrent plusieurs solutions possibles à la fois. La GA et TS approches donnent de meilleurs résultats de clustering, et ils sont moins dépendant de l'initialisation. Un inconvénient de ces approches est leur niveau record de l'exécution. Il ya beaucoup de candidats solutions, et chacun d'eux doit être affinée par les k-means algorithme. Cela rend le temps global d'exécution de manière significative supérieure à celle de la k-moyens et les méthodes sont également plus complexes à mettre en œuvre.

Dans un algorithme de recherche locale, les problèmes de conception sont:

- Représentation d'une solution.
- fonction de voisinage.
- stratégie de recherche.

3. Les algorithmes de classification non supervisée basé sur l'algorithme génétiques

3.1. L'algorithme "initialisation de K-Means utilisant les algorithmes Génétiques " [28]

3.1.1. Présentation

Le K-means est considéré comme un des majeurs de larges algorithmes utilisé dans le clustering. A cependant, il a des problèmes, et l'un de ces problèmes l'initialisation aléatoire. Un autre problème de K-means est la convergence vers le minimum local. L'algorithme génétique est un des algorithmes évolutionnaire inspirer de la nature et utilisé dans le domaine de clustering. Bashar Al-Shboul et Sung-Hyon Myaeng ont proposé deux versions pour corriger problème d'initialisation de K-means. Est ces deux versions sont : GAIK et KIGA

Ils ont proposé ses deux versions pour montrer l'efficacité du chargement de nombre des itérations du K-means aussi bien le nombre de génération des algorithmes génétiques, et pour résoudre le problème de recherche et aussi le temps de calcule.

3.1.2. L'approche GAIK

GAIK c'est un algorithme génétique basé sur le K-means, tel que l'algorithme génétique s'exécute premièrement pour donner des valeurs initiales pour le K-means, et ce dernier choisi ces valeurs pour s'exécuter d'une manière aléatoire à partir des valeurs qui a donné l'algorithme génétique, cette étape aide le K-means de sélectionné les bonnes valeurs pour trouver des bonnes solutions. La fitness appliqué est : (f(S), i, j∈ $popsize, et\ S_i > S_j$)

Figure 3. 1: le principe de l'algorithme GAIK.

3.1.3. L'approche KIGA

KIGA c'est un algorithme de K-means basé sur les algorithmes génétiques, c'est l'inverse de GAIK mais le même principe, tel que l'algorithme K-means s'exécute le premier pour initialise les solutions pour l'algorithme génétique.

Figure 3.2: le principe de l'algorithme KIGA.

3.1.4. Avantages et inconvénients

➢ ***Avantage :***

Pour l'algorithme GAIK :

- Minimise le nombre d'itération du K-means
- Converge vers la solution locale

Pour l'algorithme KIGA :

- Le temps d'exécution est réduit pour GA car K-means lui donne la population

➢ ***Inconvénient :***

Pour l'algorithme GAIK :

- Pour chaque solution initialiser de la population le processus de K-means est répété.

Pour l'algorithme KIGA :

- Le temps d'exécution de l'algorithme génétique consomme beaucoup de temps afin d'initialisé les bon centre de gravité

3.2. L'algorithme" Génétique K-means clustering pour les données mixtes" [29]

3.2.1. Présentation

Dharmendra K Roy et Lokesh K Sharma ont proposé un clustering algorithme base sur Génétique K-means, pour surmonter la limite de Génétique K-means qui est comment donné une bonne classification de cluster pour les données numérique.

On générale le problème de clustering c'est la partition des données (une donnée consiste à "n" points appartient à un espace de m-dimensionnelle dans un K différent clusters) tel que, il y a des données qui sont posé dans même cluster, mais ils sont plus similaire à un d'autre.

Les 3 problèmes de clustering sont :

1. Définir la mesure par la distance entre les différents éléments.
2. Appliquer un algorithme efficace pour déterminer les clusters des éléments les plus similaires avec aucune vérification.
3. Tirer la description pour classifier les éléments en cluster dans une brève manière.

On générale les algorithmes de clustering utilise la distance euclidienne pour juger la similarité enter deux éléments, cette technique marche bien quand la définition des attributs des données sont purement numérique dans la nature, cependant la distance euclidienne échoué de capturé la similarité entre les données quand les attributs sont mixte exemple : attributs numérique (Age, salaire…) ou catégorique exemple : homme, femme, fumé, non fumé…

3.2.2. Stratégie pour gérer les données mixtes et catégorique

Pour gérer les données mixtes et catégorique, il y a quelque stratégie sont employé. Et parmi ces stratégies : il est possible d'appliquer la distance numérique pour calculer la similarité enter les objets paire, puis convertir les valeurs attribué en catégorial et nominal à la valeur numérique, cependant, c'est très difficile de donner des valeurs numériques correct aux valeurs catégorial.

3.2.3. Description de l'algorithme génétique k-means clustering pour les données numériques mixtes et les données catégoriale

Le principe de cet algorithmes et de trouver l'optimum globale. Tous d'abord l'algorithme applique la fonction objective (fitness), puis l'opération de sélection pour sélectionner la population d'une manière aléatoire pour la prochaine génération, après il va utiliser l'opération de mutation pour faire sortir l'algorithme de minimum locale vers le minimum global, enfin il applique K-means pour converge le processus rapidement tel que il a utilisé la distance euclidien.

3.3. L'algorithme" K-means Clustering basé sur les algorithmes Génétiques" [30]

3.3.1. Présentation

L'utilisation du l'algorithme K-means est très vaste à cause de sa fiabilité, est un simple algorithme, convergence rapide et aussi peut manipuler un grand ensemble de donnée. Cependant le traditionnel K-means est sensible à l'initialisation des centres de cluster ; il fait la moyenne entre tous les objets même les centres de clusters, donc les résultats de clustering contient beaucoup de points isolés. Et pour résoudre ses problèmes, recherche des centres initiaux pour le K-means à l'aide de l'algorithme génétique. Et pour cela **Wang Min** et **Yin Siqing** ont proposé un algorithme K-means base sur l'algorithme génétique pour les deux objectifs suivant :

1) Utiliser la capacité de recherche adaptative de l'algorithme génétique pour trouver les centres initiaux de k-means.
2) Améliorer l'algorithme k-means pour réduire l'impact du point isole.

K-means algorithme est une méthode de partition de base de l'analyse typologique, mais il est sensible aux centres de classes initiaux, les centres de classes différentes générer des résultats du clustering très différentes.

Algorithme génétique est une adaptation algorithme global de recherche d'optimisation, formaté en simulant le principe de la survie du plus fort, la survie des plus aptes dans l'environnement naturel, qui comprennent principalement élection, croisement et mutation.

L'algorithme proposé dans utilise la capacité de recherche adaptative de l'algorithme génétique pour trouver les centres de classes initiaux de K-means, en suivant ces étapes :

a. La sélection de centres initiaux en utilisant l'algorithme génétique

Dans l'algorithme proposé, ils ont d'abord utilisé la fonction aléatoire pour sélectionner des objets de données K en tant que centres de classes initiaux pour former un chromosome, un total de M chromosomes sélectionnés, opérer K-means pour chaque groupe de centre de groupe dans la population initiale pour calculer la fitness, on sélectionner des individus en fonction de l'attitude de chaque chromosome, et on sélectionner haut de fitness chromosomes de l'opération de croisement et de mutation, et éliminer les chromosomes de fitness à faible, enfin formater le groupe de prochaine génération. De cette façon, dans chaque nouvelle génération de groupes, la moyenne fitness est augmentent, chaque centre de groupées plus près du centre de cluster optimale, enfin, on sélectionner les chromosomes qui ont la fitness plus forte comme le centre de classe initial.

b. Le codage du chromosome

Le codage utilisé est le codage réel, la valeur d'un gène du chromosome correspond au numéro de centre de cluster, la longueur du chromosome est le nombre de clusters, et le forme de code spécifique est : X= $(X_1, X_2 \ldots\ldots X_k)$ tel que k est le nombre de centre clusters d'un chromosome.

c. L'initialisation de la population

Sélectionner des centres de classe K au hasard pour former un chromosome R an, si le centre sélectionné au hasarda déjà existé dans le même chromosome, puis retirez le centre et sélectionnez à nouveau jusqu'à ce qu'il atteigne les centres K, jusqu'à ce que la taille de la population à M.

d. Sélection de la fonction Fitness

Utiliser l'inverse de la fonction objective J comme fonction de fitness, qui est F=1/J.

e. la Stratégie élitiste :

La stratégie élitiste dans se présente comme suit :

→ si le plus grand de fitness dans le groupe actuel est plus grand que la meilleure de fitness individuelle de la mesure, puis utiliser le meilleur individu dans le groupe actuel comme la personne la plus nouvelle jusqu'à présent.

→ sinon, remplacez le pire individu dans la génération actuelle avec le meilleur individu jusqu'à présent.

f. *Les conditions de terminaison de boucle*

L'usage de cessation d'algèbre T que l'exécution de condition de fin de l'algorithme génétique, ce qui indique que l'arrêt de l'algorithme génétique en cours d'exécution à l'algèbre évolution spécifié, et délivrer en sortie le meilleur individu dans le groupe actuel comme solution optimale du problème.

g. *Description de l'algorithme spécifique*

1) Définir les paramètres: taille de la population, le nombre maximal de T itération, le nombre de clusters K…etc.
2) Générer aléatoirement m chromosomes, un chromosome représente un ensemble de centres de classes initiaux, afin de former la population initiale.
3) Selon les centres de classes initiaux ont montré par chaque chromosome, effectuer K-means, chaque chromosome correspond à une fois K-means, puis de calculer le chromosome de fitness en ligne avec un résultat de clustering, et mettre en œuvre la stratégie de conservation optimale.
4) Pour le groupe, d'effectuer la sélection, de croisement et opérateur de mutation pour produire une nouvelle génération de groupe.
5) Afin de déterminer si ces conditions respectent les conditions de résiliation génétiques, si elle rencontre alors opération de retrait génétique et tournez 6, sinon mettez 3.
6) Calculer remise en forme de la nouvelle génération du groupe, à comparer avec l'aptitude du meilleur individu dans le groupe actuel de remise en forme avec le meilleur individu jusqu'à présent de trouver la personne avec le plus haut de fitness.
7) Réaliser des K-means selon le centre de l'amas initial représenté par le chromosome avec la plus grande remise en forme, puis résultat de clustering de sortie.

3.4. Hybridation entre l'algorithme génétique (GA) et PSO [31]

3.4.1. Présentation

Cet article propose une nouvelle technique de clustering (HGAPSOA), basé sur une hybridation entre l'algorithme génétique (GA) est l'algorithme d'essaim de particule (PSOA) pour le problème de clustering. Tel que, cette technique intègre les essaims particules d'optimisation, l'algorithme génétique et la méthode de K-means.

D'apprêt les tests avec les différents benchmarks (Iris, Glass, Vowelet Wine), les résultats montrent que l'algorithme proposé est plus efficace que GA et PSO.

3.4.2. *L'algorithme HGAPSOA*

L'algorithme HGAPSOA initialise aléatoirement une population de taille 2N. Ces individus peuvent être considérés comme des chromosomes dans l'algorithme génétique, ou comme des particules dans PSO. 2N individus sont introduits dans l'algorithme GA pour créer 2N nouveaux individus par reproduction, croisement et l'opérateur de mutation. Ensuite, les 2N individus sont triés par fitness, et les n individus sont appliqués pour l'amélioration du PSOA. A la fin de HGAPSOA, le K-means est utilisé pour le but de converge vers la solution optimale rapidement.

La figure 3.3 représente est une optimisation par essaim de particules. Tel que, dans l'amélioration du PSOA, il y a un crossover pour chaque particule. Ensuite, nous pouvons obtenir deux particules filles. Après la comparaison, la particule avec la valeur de fitness la plus petite soit l'enfant finale.

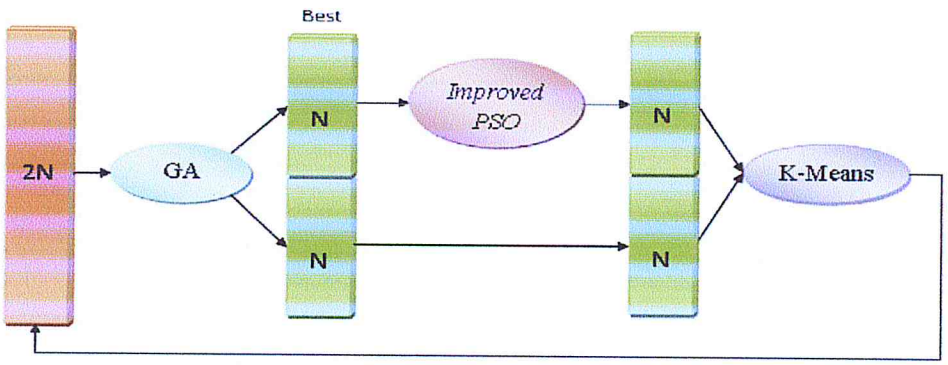


Figure 3. 3: La structure de la technique HGAPSOA.

3.4.3. *Fonctionnements de la technique HGAPSOA*

Le fonctionnement de la technique HGAPSOA suit les étapes suivantes :

- ***Étape 1 :*** Configurez les paramètres tels que la taille 2N de la population (nombre d' particules), vitesse maximale (V-MAX)…
- ***Étape 2 :*** Initialisé chaque particule au hasard avec la position initiale, et vitesse. Dans PSO clustering, nous avons besoin de connaître le nombre des grappes K à l'avance.
- ***Étape 3 :*** Calculer la valeur de fitness.
- ***Étape 4 :*** Appliquer les opérateurs de l'algorithme GA (reproduction, croisement et mutation) pour la population 2N, et créer un autre 2N population.
- ***Étape 5 :*** Évaluer la fitness pour chacun des individus 2N. Classez-les par la valeur de fitness.
- ***Étape 6 :*** Mettre à jour la position de meilleur global et la meilleures locale.
- ***Étape 7 :*** Mise à jour de la vitesse.
- ***Étape 8 :*** Mise à jour le vecteur de position.
- ***Étape 9 :*** Appliquer les opérateurs de K-means.
- ***Étape 10 :*** Si l'un des critères d'arrêt est satisfait passez à l'étape 12. Sinon, passez à l'étape 3.
- ***Étape 11:*** La sortie de la particule avec une valeur minimum de fitness dans la dernière génération.

3.5. Quelques hybridations de PSO avec d'autres méthodes [32]

Clustering gène est l'un des plus populaires applications dans le domaine de la bio-informatique. Il s'agit d'une méthode de regroupant gène en grappes, de sorte que chaque groupe doit avoir les niveaux d'expression des gènes similaires. Les deux plus populaires algorithmes de recherche sont : les essaims de particules (PSO) et algorithme génétique (GA). Ces deux algorithmes sont utilisés pour regrouper des données d'expression de gènes, mais le principal inconvénient de ces deux algorithmes c'est qu'ils furent pris au piège des optimums locaux tout en explorant l'espace de problème. Ainsi, le PSO et GA ont été hybridés avec plusieurs autres méthodes, afin de surmonter le problème des optimums locaux. Cet article présente une littérature enquête à la fois GA et la demande de PSO dans le clustering de gène.

3.5.1. L'hybridation de PSO avec K-means

K-means est un algorithme classique, et l'un des populaires méthodes. Mais le principal inconvénient de K-means est qu'il est facilement obtenu le mauvais optimum local. Afin d'améliorer les performances de K-Means, Zhihua. et. al ont proposé une forme modifiée de K-means algorithme connu sous le nom PK-Means qui est le combinaison d'essaim de particules et K-Means algorithme de clustering. Les résultats de la PK-Means montrent que cet algorithme possède un taux de convergence plus rapide et peu charge de calcul et surpasse également le précédemment algorithmes connus k-means.

Yau roi Lam propose un nouvel algorithme XK-moyens qui s'hybrident PSO avec K-means. La méthode proposée est comparée avec l'algorithme K-means et l'algorithme PK-Means. Les observations suivantes sont rapportées. Tout d'abord, la compacité de la grappe et la stabilité montrent des meilleurs résultats que ceux obtenus à partir de K-means et les algorithmes PK-moyens. Deuxièmement, la méthode proposée est au moins trois fois plus vite et enfin, la complexité du l'algorithme est faible en comparaison de K-means et PK-moyens algorithme.

3.5.2. L'hybridation de PSO avec l'algorithme génétique

Enrique. Compare l'utilisation d'une particule d'essaim (PSO) et un algorithme génétique (GA) pour classifier les données de grande dimension. Tous les deux les algorithmes sont utilisés pour trouver des échantillons d'information gènes de milliers de gènes. L'algorithme proposé PSO/GA trouve des gènes très intéressants. Tel que, cette approche a été testé sur six ensembles de données sur le cancer. Et les résultats de l'algorithme proposé aident beaucoup dans l'identification des gènes spécifiques.

Comme les données d'expression génique de clustering basés sur la mesure de similarité est devenu l'un des importants domaines de recherche dans le domaine de la bio-informatique, Shutao. A proposé une méthode de sélection des gènes en utilisant une hybridation entre PSO avec GA. La proposition hybride PSO / GA est adoptée pour sélectionner des sous-ensembles de gènes les plus importants. L'algorithme a été essayé sur les plusieurs ensembles de données d'expression de gènes. Les résultats expérimentaux montrer une amélioration dans la formation du groupe de gènes et conduit également à une meilleure précision de regroupement.

4. Conclusion

Le domaine de la bio-informatique nécessite l'informatique pour le traitement de grandes quantités de données et divers problèmes tels que Gene Clustering problème, Problème d'accueil moléculaire, construction de l'arbre phylogénétique, l'ARN secondaire Structure Prédiction etc. [32]

Clustering devient populaire de nos jours en raison de maturité technologique et puissance de l'informatique augmentant. Bien qu'il existe de nombreux algorithmes de classification développée, K-means est l'un des méthodes populaires pour le clustering en raison de son efficacité et sa simplicité. Mais il présente l'inconvénient de choisir le centre de gravité initial et facilement se coincer dans l'optimum local.

Ainsi, l'algorithme basé sur la population comme PSO et GA, ont récemment acquis une grande popularité parmi les chercheurs, pour leur capacité à trouver une solution optimale robuste. GA et PSO et d'autres algorithmes ont été hybridés avec d'autres métaheuristiques par plusieurs chercheurs, pour régler le problème d'initialisation, gestion de clusters et la convergence vers le minimum local.

Chapitre 4:

La modélisation d'un algorithme intelligent

«L'imagination est plus importante que le savoir. »

Albert Einstein

1. Introduction

Clustering c'est un processus de groupement de données dans des clusters, d'où les objets dans chaque cluster ont une grande similarité, et sont différents aux objets qui sont dans les autres clusters. Clustering est utilisé en plusieurs domaines, inclure la fouille de données, statistiques, biologie et l'apprentissage de machine …etc.

Parmi les différents algorithmes de clustering, K-means qui est une méthode la plus populaire utilisée pour l'analyse des données. Cependant, on observe que le KM converge vers l'optimum local, et ses résultats dépendent de l'initialisation qui est généralement aléatoire. Tel que chaque exécution de KM sur les mêmes données produit des différents résultats.

L'algorithme des abeilles (**BSO** : **B**ees **S**warm **O**ptimisation) tentative d'incorporer les idées d'évolution naturel. Généralement, il commence d'initialiser aléatoirement la solution référence à partir d'une population donnée, Après, chaque abeille de la colonie explore sa région avec l'opération calcul de voisinage. Chaque abeille retourne le meilleur voisin en mettant ce dernier dans la table dance. La meilleure solution de la table dance devient la solution référence pour la prochaine itération.

Dans ce chapitre, on établit une modélisation des approches proposées qui sont basées sur l'algorithme BSO.

2. Les opérations principales de BSO

Les opérations principales de colonie des abeilles sont : détermination Search Area et calcule de voisinage. Dans cette partie on va proposer plusieurs stratégies de recherche qui vont être appliquées par la suite dans les approches proposées.

2.1. Détermination Search Area

Le but principal de cette opération est de déterminer les différentes régions de recherche qui sont le point de départ de chaque abeille de la colonie. Pour cela, on propose plusieurs stratégies afin de sélectionner la meilleure stratégie dans l'expérimentation. La détermination se fait à partir de la solution référence nommée Sref. Les stratégies de recherches sont les suivantes:

2.1.1 Stratégie SAUT AVEC FLIP

Premièrement, on positionne un conteur "i" dans Sref, et chaque fois on incrémente le « i » et on l'ajoute FLIP qui doit être sélectionné par l'utilisateur. On répète ce processus pour tous les éléments de la solution Sref multiple de FLIP.

a) **Exemple :** Soit FLIP=2, $B_1 \ldots B_k$ K abeilles, et la solution référence Sref= (5,20, 9, 13, 7, 14, 1).

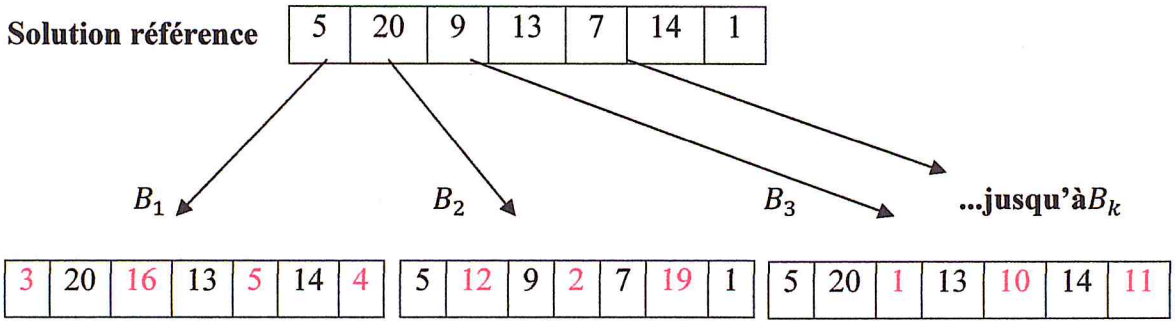


Figure 4. 1: Illustration de la stratégie Saut avec Flip.

La Figure (4.1) illustre la construction des abeilles en utilisant la stratégie Saut avec Flip. Chaque abeille « i » commence à partir de la i^{éme} case de Sref de telle sorte qu'on modifie les cases (i, i+2, i+4….).

b) **Algorithme**

```
i, j, n, FLIP: entier ; x: réel ; T : tableau(50) réel
Début
    Lire(n) ; lire(FLIP)
    Pour i=0 à n faire
        J=i+FLIP ;
        Tant que j<n faire
            x= nbr_aléatoire(30) ; T(j)= x;
            j=j+FLIP
        Fait;
    Fait;
Fin
```

c) Analyse

à partir de la solution référence, la stratégie du SAUT AVEC FLLIP permet de déterminer les différentes régions de recherche, au début ces régions sont éloignées car à chaque fois on saute d'une case à une autre la valeur change, donc la distance entre les abeilles sera grande, mais les régions trouvées seront semblable à la solution référence avec une différence entre deux cases ou une, et ça implique une très petite distance entre les abeilles donc les régions de recherche seront très petites (proches). Comme on remarque, si le FLIP est petit, la distance entre abeilles est très grande donc les régions trouvées sont grandes impliquent des régions éloignées. Par contre si le FLIP est grand la distance entre les abeilles sera petite donc les régions trouvées sont plus petites impliquent des régions proches.

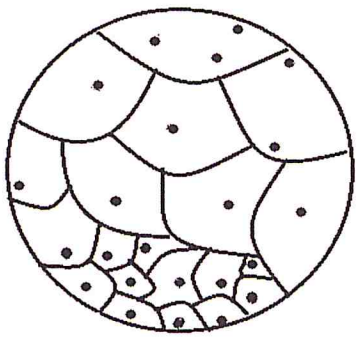


Figure 4. 2: Illustration de la Stratégie SAUT AVEC FLIP.

2.1.2 Stratégie SUIVANT AVEC FLIP

Le fonctionnement de cette stratégie est basé sur le FLIP. Premièrement, l'utilisateur donne une valeur entier au FLIP et à partir d'un tableau Sref, on met un compteur « i » dans une case, et on change les valeurs des cases successives, tel que le nombre de ces cases est égale au FLIP c.à.d. si le FLIP=4 on change les valeurs 4 cases.

a) Exemple : Soit FLIP=4

La Figure (4.3) illustre la construction des abeilles en utilisant la stratégie Suivant avec Flip. Chaque abeille « i » commence à partir de la $i^{éme}$ case de Sref de telle sorte qu'on modifie les cases (i, i+1, i+2….).

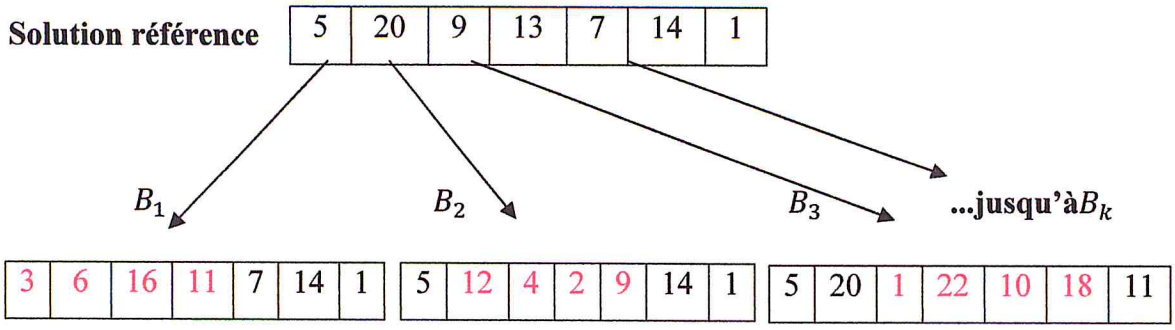


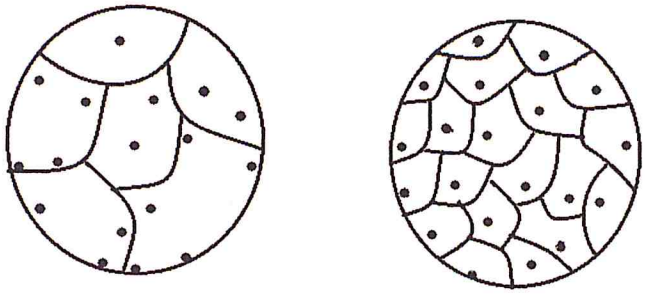
Figure 4. 3: Illustration de la stratégie Suivant avec Flip.

b) Algorithme

```
i, j, n, FLIP: entier ; x: réel ; T : tableau(50) réel
Début
    Lire(n) ; lire(FLIP)
    Pour i=0 à n faire
        Pour j=i à FLIP faire
            x= nbr_aléatoire(30) ;
            T(j)= x ;
        Fait;
    Fait;
Fin
```

c) Analyse

D'après l'exemple et l'algorithme, on remarque que cette stratégie ne donne jamais les mêmes régions, aussi elle génère des régions éloignées donc la distance entre les abeilles est grande, et équilibrées c.à.d. des régions ont presque la même taille. Cependant, si le nombre de FLIP est petit, la distance entre abeilles sera petite donc les régions trouvées sont plus petites qui implique des régions proches.



(a) (b)

Figure 4. 4: Illustration de la Stratégie SUIVANT AVEC FLIP, (a)FLIP est grand et (b) le FLIP est petit.

2.1.3 Stratégie HYBRIDE

Cette stratégie est composée des deux stratégies précédentes **SAUT AVEC FLIP** et **SUIVANT AVEC FLIP**. Son principe consiste de deviser le tableau Sref en deux parties. De telle sorte, dans la première partie du tableau, on applique SAUT AVEC FLIP, par contre, pour la deuxième partie de tableau, on applique la stratégie SUIVANT AVEC FLIP.

a) **Exemple** : soit FLIP=3

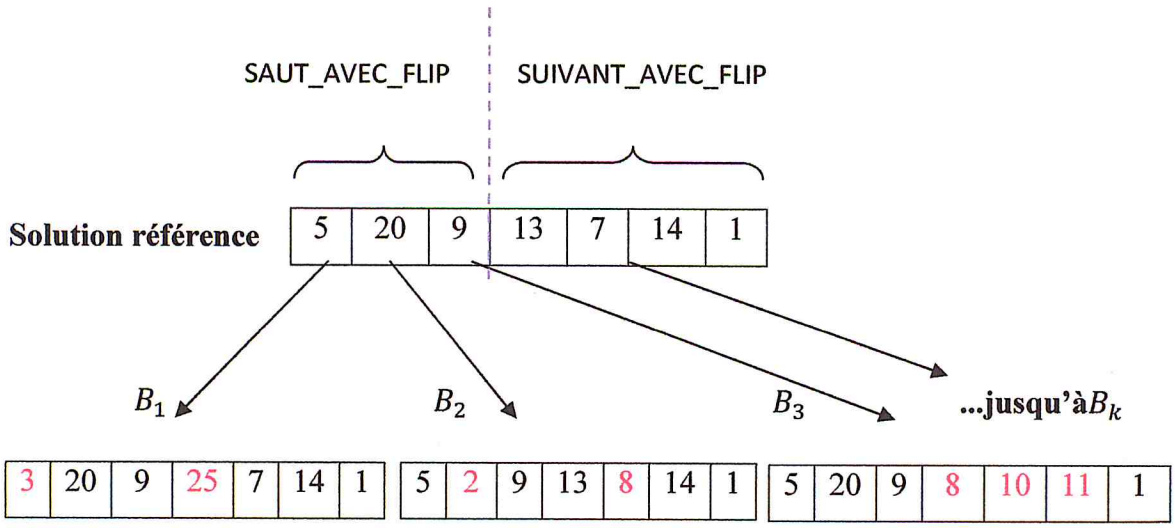


Figure 4. 5: Illustration de la stratégie HYBRIDE.

b) Algorithme

```
i, j, n, FLIP: entier ; x: réel ; T : tableau(50) réel
Début
    Lire(n) ; lire(FLIP)
    Pour i=0 à n/2 faire
        j= i+ FLIP ;
        Tant que j<n faire
            x=nbr_aléatoire(30) ;T(j)= x;  j=j+FLIP
        Fait;
    Fait;
    Pour i=n/2+1 à n faire
        Pour j=i à FLIP faire
            x= nbr_aléatoire(30) ;   T(j)=x ;
        Fait;
    Fait;
Fin
```

c) Analyse

L'hybridation entre la stratégie SAUT AVEC FLIP et SUIVNT AVEC FLIP permet de faire une équivalence entre les régions, tel que la distance entre les abeilles dépend de FLIP. Le but d'appliquer le SAUT AVEC FLIP en premier c'est pour éviter les petites régions à la fin.

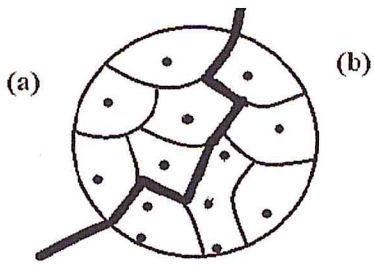


Figure 4. 6: Illustration de la Stratégie HYBRIDE, (a) la stratégie SAUT AVEC FLIP, (b) la stratégie SUIVNT AVEC FLIP.

2.2 Les voisinages

On appelle voisinage d'une solution donnée S l'ensemble des solutions qui peuvent être générées à partir de cette solution. Par exemple, on peut inverser les cases de la solution, on change la valeur d'une case etc. Pour cela, on propose deux stratégies, le but principal de ces stratégies est de trouver tous les solutions les plus proches:

2.2.1 Stratégie JUMP

Le principe de cette stratégie est basé sur le JUMP. Tel qu'on saute d'une manière aléatoire d'une case à une autre dans un tableau donné, en changeant à chaque fois la valeur de la case visitée.

a) **Exemple** : le JUMP est un nombre entier choisi aléatoire entre 0 et 6.

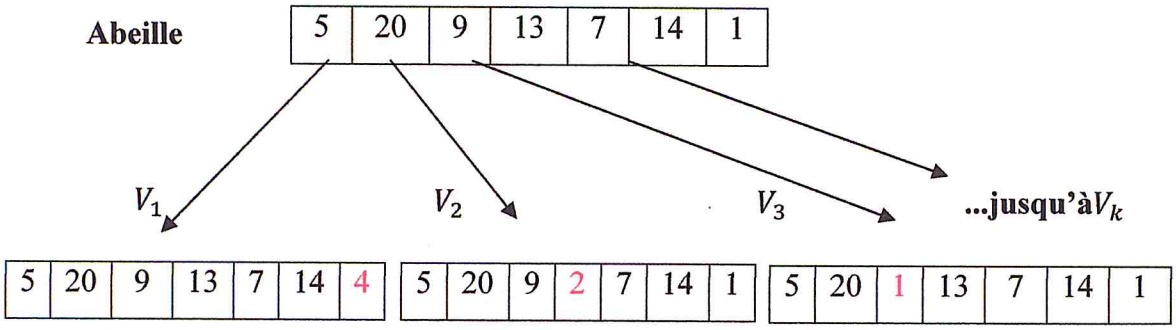


Figure 4. 7: Principe de la stratégie JUMP.

b) **Algorithme**

```
i, n, JUMP: entier ; x: réel ; T : tableau(50) réel
Début
    Lire(n) ;
    Pour i=0 à n faire
        x=nbr_aléatoire(30) ;
        JUMP= nbr_aléatoire(n);   T(JUMP)=x ;
    Fait;
Fin
```

c) **Analyse :** La stratégie de COUNTER donne tous les voisinages d'une solution, avec aucune répétition.

3. Les approches proposées

3.1. *Approche "1": K-means basé sur le BSO (KBSO)*

Cette approche a pour but de converger vers le minimum globale, son principe est d'exécuter l'algorithme des abeilles en premier à partir d'une population donnée afin de transmettre les meilleurs K éléments de centre de gravité au K-means, en faisant un bon regroupement des objets.

La modélisation de cette approche suit les étapes suivantes :

→ Création de la solution référence.
→ Codage de la solution.
→ Fonction d'évaluation fitness.
→ Déterminer le search area.
→ Calculer de voisinage.

- **Création de la solution référence :** On va choisir aléatoirement des valeurs réels [1…n] pour chaque centre de gravité de la solution référence. En générale la solution est un vecteur de K éléments, et chaque élément il a un nombre d'attributs.

- **Codage de la solution :** Pour le codage, on va utiliser le codage réel, pour que tous les objets soient en même type.la forme du codage est un vecteur de K éléments tel que chaque élément est un nombre réel compris entre 1 à n.

  $X = (X_1, X_2, X_3, \ldots\ldots X_n)$

  X est un élément.

  n est le nombre de centre de gravité.

- **Fonction d'évaluation fitness :** dans cette approche, on a utilisé la formule de fitness suivante :

$$\min^{f}(s) = \sum_{\substack{i=1 \\ j=1}}^{k} |D(s[i], s[j])| \text{ avec } i \neq j.$$

S : c'est la solution (c'est un vecteur de k centres).

$D(s[i], s[j])$ **:** C'est la distance entre les éléments de la solution S (distance de Manhattan).

a) Exemple : soit la solution S (3, 6, 2, 1)

S	3	6	2	1

Donc la fitness est comme suit :

F(S)= |3-6|+|3-2|+|3-1|+|6-2|+|6-1|+|2-1|=3+1+2+4+5+1=16 Alors F(S) = 16.

b) Algorithme

```
i, j, n: entier ; F: réel ; S : tableau(50) réel
Début
    Lire(n) ; F=0 ;
    Pour i=1 à n faire
        Pour j=i+1 à n faire
        F= F+S(i) - S(j) ;
            Si F<0 alosr
            F=F*(-1);
            Fsi;
        Fait;
    Fait;
Fin
```

Pour déterminer le "Search Area" et le Calculer de voisinage, on va choisir une des stratégies proposées précédemment. La Figure (4.9 et 4.10) explique clairement l'architecture de cette approche.

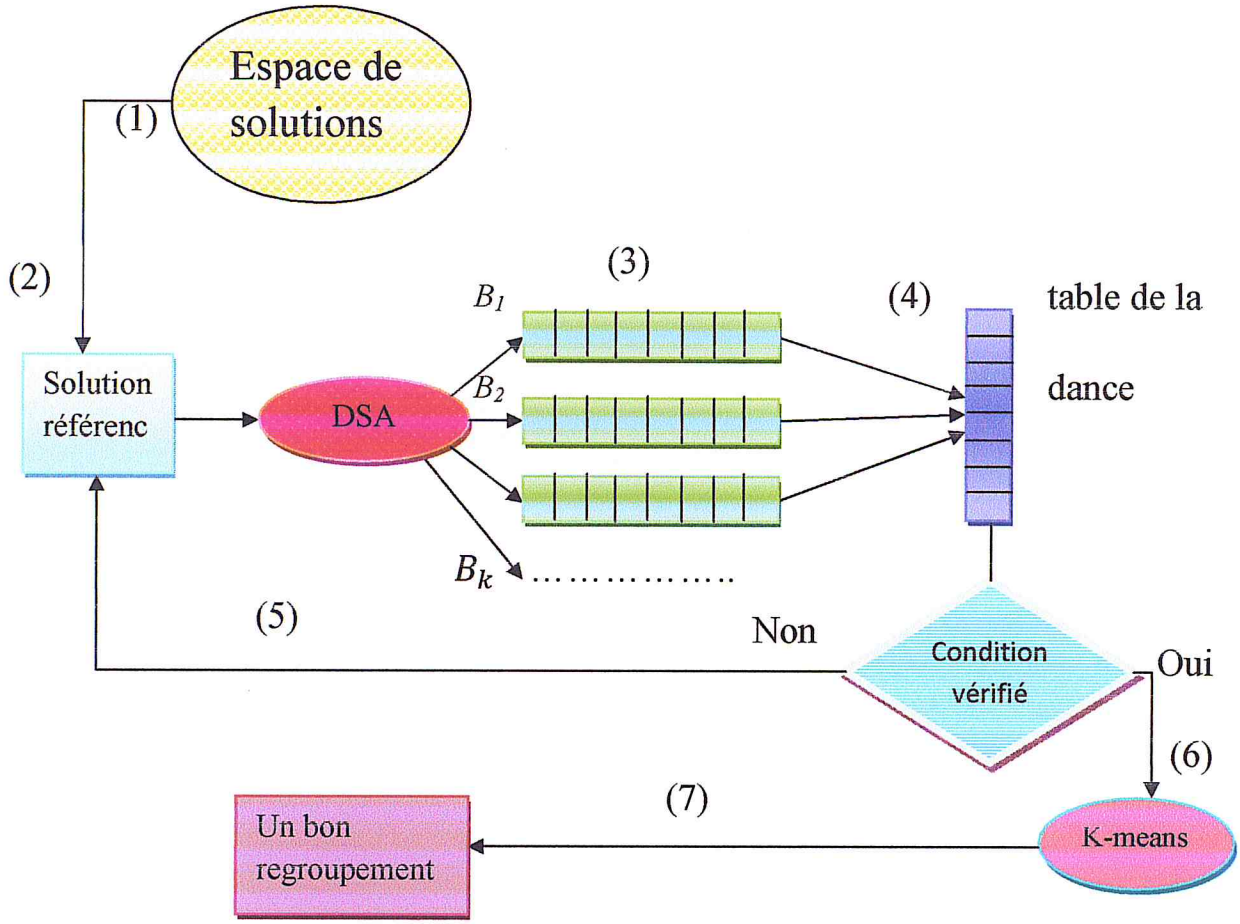


Figure 4. 9: structure de la première approche.

c) Explication

(1) L'initialisation aléatoire de la solution référence à partir d'espace de solution.

(2) Déterminer le Search Area(DSA), tel que $B_1 \ldots B_k$ représente les abeilles.

(3) Calculer les voisinages de chaque région.

(4) Extraire la solution minimale pour chaque région et les posées dans la table de dance.

(5) Si la condition n'est pas vérifiée alors la solution trouvée sera à la place de la solution référence.

(6) Sinon, la solution trouvée sera le K centre initiaux de l'algorithme K-means.

(7) Exécution de l'algorithme K-means.

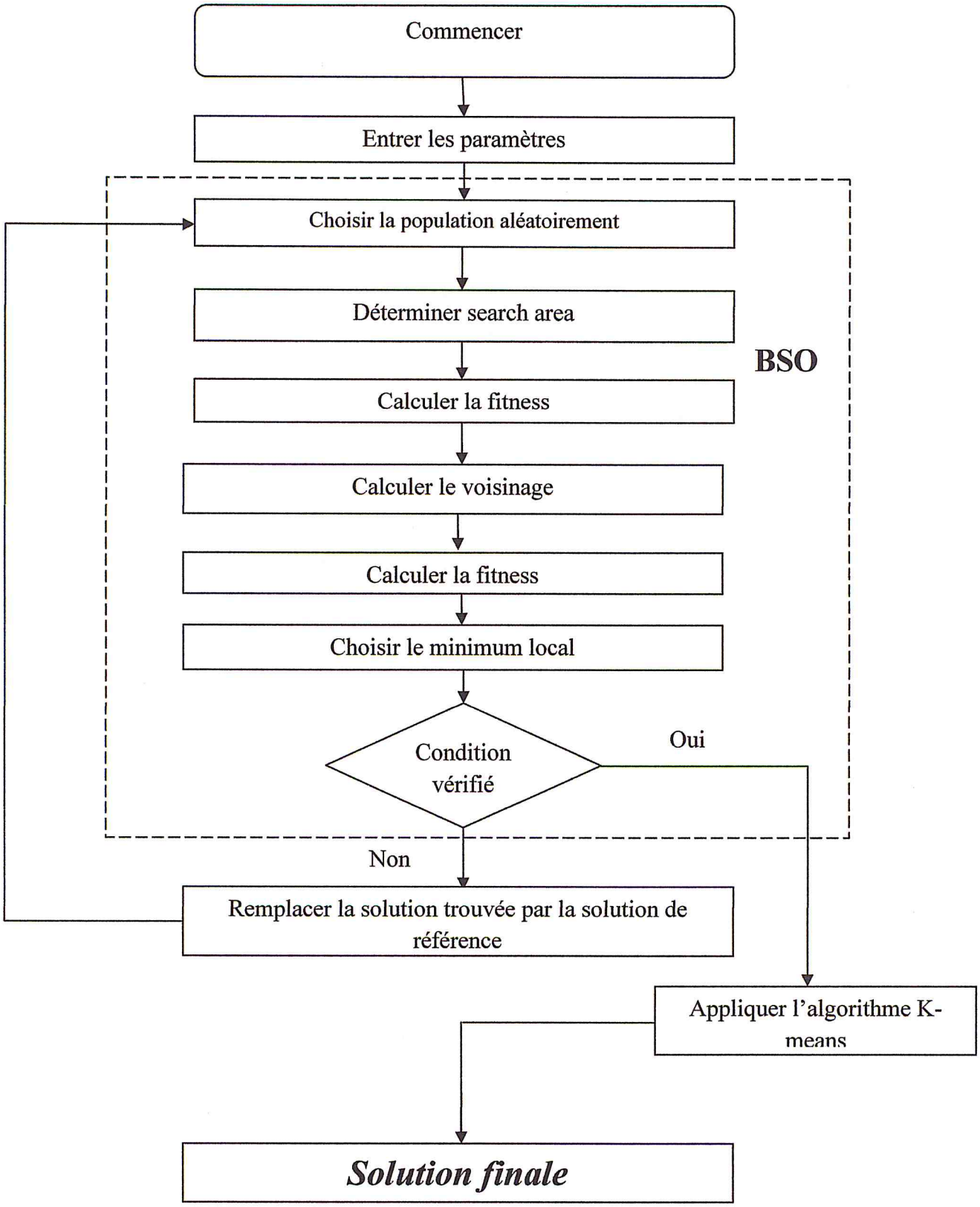


Figure 4. 10: Organigramme de la première approche.

Explication

(1) L'initialisation aléatoire de k centres de gravité.
(2) Exécution de l'algorithme K-means.
(3) Affecter les k centres trouvées par K-means à la solution référence de BSO.
(4) Déterminer le Search Area(DSA), tel que $B_1 \ldots B_k$ représente les abeilles.
(5) Calculer les voisinages de chaque région.
(6) Extraire la solution minimale pour chaque région et les posées dans la table de dance.
(7) Si la condition n'est pas vérifiée alors la solution trouvée sera à la place de la solution référence.
(8) Sinon, la solution trouvée sera la solution finale de l'algorithme.

4. Conclusion

Le K-means est un algorithme de clustering le plus utilisé et le plus connu, mais le choix aléatoire du centre initiale du cluster implique l'instabilité des résultats de clustering. D'autre part la métaheuristique BSO montre son efficacité dans plusieurs domaines tels que la recherche d'information.

Dans ce chapitre, nous avons proposé deux approches qui sont basées sur les algorithmes K-means et BSO. Dans la première approche, on initialise les centres de gravité de K-means à l'aide de BSO, par contre dans la deuxième approche, on initialise la solution référence de BSO en appliquant K-means en premier. Ces deux approches élimine la sensibilité de l'initialisation des données pour l'algorithme K-means traditionnel, et obtenir des clusters plus stables et avec une haute qualité.

Dans le chapitre suivant, on va faire une comparaison expérimentales des approches proposées afin d'extraire la meilleure approche.

Chapitre 5 :

Implémentation et expérimentation

«Le commencement de toutes les
sciences, c'est l'étonnement de ce que
les choses sont ce qu'elles sont. »

Aristote

1. Introduction

Nous allons voir dans ce chapitre l'implémentation en java des deux approches, qui ont étéproposéesprécédemment dans le quatrième chapitre, ainsi de ces expérimentations de la mise en œuvre de ces dernières sur des benchmark généré aléatoirement et de différentes taille,afin d'en déduire la meilleure approche parmi les approches proposées.

2. Implémentation de l'application BSOCK

Voici l'interface principale de l'application **BSOCK: BSO Clustering based K-means :**

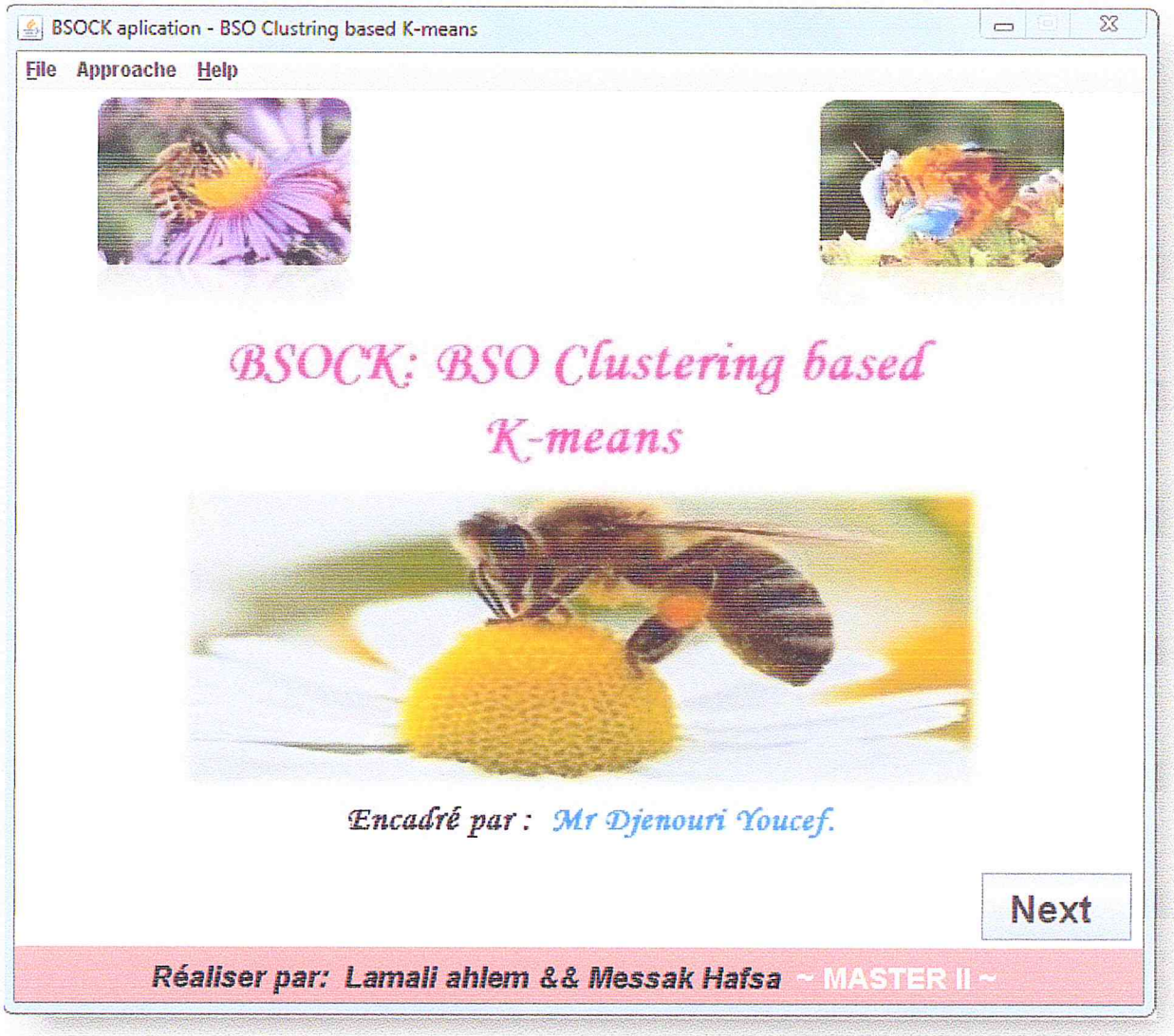


Figure 5. 1: Interface de l'application BSOCK.

Derrière cette interface il y'a les classes suivantes :

2.1. Les classes communes pour les deux approches

- ***La classe center***

```
Public class center {
float centre [] ;}
```

C'est une classe qui permet de représenter la structure des attributs.

- ***La classe solution***

```
Public class solution {
center sol [] ;
float cout; }
```

C'est une classe qui permet essentiellement de représenter la structure de la solution telle que la solution est de type center et aussi le cout qui représente la valeur de meilleur fitness en utilisant la formule suivante que nous avons vu dans le quatrième chapitre :

$$\min^{f}(s) = \sum_{\substack{i=1 \\ j=1}}^{k} |D(s[i], s[j])|$$

➢ *La classe objet*

```
public class objet {

center obj;

int cluster;}
```

C'est une classe qui permet de représenter l'objet de benchmark qu'on va utiliser avec un paramètre cluster qui permet d'assigner le cluster.

2.2. Les classes de la première approche : « K-means base BSO »

Appart les classes communes, on aura trois autres classes :

➢ *La classe BSO*

```
public class BSO
{
int flip; // paramètre dans les stratégies de recherche
int k; // le nombre de clusters
int n; // le nombre d'attributs
solution Tbees[]; // tableau d'abeilles
solution neighbors[]; // tableau de voisins
solution Sref; // la solution référence

void init()
void donwload()
void fitness(solution s)
void SAF(solution T, int flip, int n)
void NAF(solution T, int flip, int n)
void Hybrid(solution T, int flip, int n)
void jump(solution bee)
void parcourir(solution bee)

}
```

Cette classe est constituée essentiellement des méthodes suivantes :

- **init ():** Cette méthode initialise les structures de données suivantes :(Sref, Tbees, neighbors).
- **donwload ():** Cette méthode récupère le tableau objet.
- **fitness ():** Cette méthode calcul la fitness de chaque solution.
- **SAF ():** Cette méthode applique la stratégie de search area « Saut Avec Flip » .Elle permet de déterminer les différentes régions de recherche, à chaque fois quand on saute d'une case à une autre, la valeur va changer aléatoirement.
- **NAF ():** Cette méthode s'appelle « Suivant Avec Flip », elle permet de changer la ou bien les valeurs des cases successives en fonction du paramètre flip.
- **Hybrid ():** Elle permet de fusionner les deux stratégies **SAF** et **NAF,** tel que dans la première moitié du notre tableau, on applique **SAF**, et pour la deuxième, on applique la stratégie **NAF.**
- **jump ():** Cette méthode nous permet de calculer le voisinage, tel qu'on fait le saut d'une manière aléatoire d'une case à un autre en changeant sa valeur.
- **parcourir () :** Elle applique la stratégie de **COUNTER.** On parcoure tous les éléments de solutions en changeant à chaque fois la valeur rencontrée.

➢ *La classe k- means*

```
public class  kmeans
{
Int M; // le nombre d'objets
Int k; // le nombre de clusters
Int n; // le nombre d'attributs
objet T_objet [];
solution centers;

void  init ()
void  donwload_dataset ()
void  initialiser_centers ()
void  display_dataset_and_centers ()
float distance (objet x, center y)
void  new_center_computation (int cluster)
void  assigned_clusters ()
void  kmeans_function ()

}
```

Cette classe est constituée des méthodes suivantes :

- **init ():** Cette méthode initialise les structures de données suivantes :(centers and T_objets).
- **donwload_dataset() :** Elle récupère les donnes à regrouper et le mettre dans **T_objet**.
- **initialiser_centers () :** Cette méthode initialise les centres de gravite des k clusters en affectant aléatoirement.
- **display_dataset_and_centers () :** Cette méthode permet d'afficher les objets et les centres final de l'algorithme k-means.
- **distance () :** Cette méthode calculer la distance entre deux objets.
- **new_center_computation () :**
- **assigned_clusters ()** : Elle affecte les objets dans les bon clusters.
- **kmeans_function1 () :** Elle Définit l'algorithme principale de K-means.

➢ *La classe principale « main »*

```
public class approche1 {
     void process() {

      int IMAX=20;
      solution meilleur;
      meilleur=new solution();
      BSO b=new BSO();
      b.init();
      b.creer();
     int iteration=0;
      while(iteration<IMAX) ;
      {
      b.NAF(b.Sref,b.flip,b.n);
      for(int i=0; i<b.Tbees.length;i++)
      {
       b.parcourir(b.Tbees[i]);
      }
      int i=0;
      meilleur=b.Tbees[0];
      for (i=1; i<b.k;i++)
      {
       if (meilleur.cout>b.Tbees[i].cout)
       {
           meilleur=b.Tbees[i];
       }

      }
      b.Sref=meilleur;
      iteration++;
      }
   Kmeans k=new Kmeans();
   k.Kmeans_function1(meilleur);

    }
    }
```

2.3. les classes de la deuxième approche : « BSO base K-means »

Dans cette approche on a utilisé les mêmes classes précédentes, tel que dans la classe « **main** », on fait le travail de façon inverse :

➢ ***La classe principale << main >>***

```
public class approche2 {
void process(){
        int iteration=0; ; solution meilleur;
        int IMAX=20;
        Kmeans k=new Kmeans();
        k.init();
        BSO b=new BSO();
        b.init();
        b.Sref=k.Kmeans_function2();
        while(iteration<IMAX)
            {
            b.NAF(b.Sref,b.flip,b.n);
            for(int i=0; i<b.k;i++)
            {
             b.parcourir(b.Tbees[i]);
            }
            int i=0;
            meilleur=b.Tbees[0];
            for (i=1; i<b.k;i++)
            {
                if (meilleur.cout>b.Tbees[i].cout)
                {
                    meilleur=b.Tbees[i];
                }
             }
            b.Sref=meilleur;
            iteration++;
            }
```

3. Expérimentation de l'application BSOCK

3.1. Réglage de paramètre

Dans cette partie, on essaye de régler les paramètres des approches proposées comme suit:

a. La première approche

En fixant à chaque fois le nombre de données à regrouper (M=10000 données avec N=20), on varie IMAX (nombre d'itérations), (k : le nombre d'abeilles et de clusters), de la première approche. On aura les résultats de la fitness comme suit:

K	IMAX	*Le Résultat de la fitness*		
		NAF	SAF	Hybride
20	20	**25**	**26**	**28**
20	40	**30**	**25**	**30**
20	60	**20**	**23**	**25**
20	80	**24**	**20**	**30**
20	100	**20**	**20**	**20**
40	20	**15**	**15**	**15**
40	40	**15**	**12**	**14**
40	60	**16**	**18**	**19**
40	80	**14**	**12**	**11**
40	100	**15**	**10**	**10**
60	20	**12**	**12**	**9**
60	40	**12**	**12**	**8**
60	60	**11**	**10**	**7**
60	80	**11**	**11**	**5**
60	100	**11**	**11**	**5**
80	20	**10**	**9**	**5**
80	40	**10**	**9**	**7**
80	60	**10**	**10**	**8**
80	80	**10**	**11**	**15**
80	100	**9**	**12**	**12**
100	20	**9**	**24**	**24**

100	40	**18**	**25**	**15**
100	60	**18**	**28**	**14**
100	80	**19**	**28**	**15**
100	100	**28**	**38**	**30**

Tableau 5. 1: les résultats de la fitness de la première approche.

On remarque à partir de ce tableau que la solution engendré s'améliore continuellement tout en augmentant le nombre de clusters et le nombre d'itérations ainsi la stratégie HYBRIDE est la meilleure stratégie de ces trois. En plus de ça, elle trouve un regroupement de distance de 5 pour 60 abeilles et 80 itérations. On Peut dire que la meilleur configuration de cette approche est : **Stratégie = Hybrid, IMAX = 80 et K= 60.**

b. La deuxième approche

Comme dans la première approche, on fixe le nombre de données à regrouper (M=10000 données avec N=20), et on varie IMAX (nombre d'itérations), (k : le nombre d'abeilles et de clusters), de la deuxième approche. On aura les résultats de la fitness comme suit:

K	IMAX	*Le Résultat de la fitness*		
		NAF	SAF	Hybrid
20	20	**35**	**36**	**38**
20	40	**25**	**30**	**28**
20	60	**26**	**23**	**20**
20	80	**24**	**20**	**25**
20	100	**20**	**20**	**20**
40	20	**18**	**19**	**17**
40	40	**17**	**19**	**18**
40	60	**18**	**18**	**19**
40	80	**18**	**12**	**14**
40	100	**19**	**10**	**13**
60	20	**18**	**13**	**12**
60	40	**17**	**14**	**11**
60	60	**15**	**12**	**11**

60	80	**12**	**11**	**11**
60	100	**10**	**10**	**10**
80	20	**11**	**8**	**9**
80	40	**12**	**7**	**8**
80	60	**13**	**10**	**6**
80	80	**14**	**12**	**5**
80	100	**14**	**11**	**4**
100	20	**16**	**14**	**8**
100	40	**18**	**15**	**10**
100	60	**19**	**17**	**14**
100	80	**20**	**25**	**19**
100	100	**35**	**40**	**20**

Tableau 5. 2: les résultats de la fitness de la deuxième approche.

Comme la première approche, on remarque à partir de ce tableau que la solution engendré s'améliore continuellement tout en augmentant le nombre de clusters et le nombre d'itérations ainsi la stratégie HYBRIDE est la meilleure stratégie de ces trois. En plus de ça elle trouve un regroupement de distance de 4 pour 80 abeilles et 100 itérations. On Peut dire que la meilleur configuration de cette approche est : **Stratégie = Hybrid, IMAX = 100 et K = 80.**

3.2. Expérimentations

Après le réglage de paramètre des deux approches, on teste les deux approches avec des benchmarks généré aléatoirement de différentes tailles. En augmentant la taille de données à regrouper et en fixant le nombre d'attributs à 100. On obtient les résultats suivants :

Dans les tableaux suivants, chaque case remplie est le résultat de la moyenne de cent (100) essais consécutifs.

a. L'approche « K-means base BSO »

En fixant K à 60, IMAX à 80, stratégie de détermination de Search Area à Hybride. Et on varie le calcul de voisinage et la taille de données. On aura les résultats suivants :

Taille de benchmark	La meilleure fitness obtenue		Le Temps d'exécution en (SEC)	
	Jump	Counter	Jump	Counter
10000	5	8	1200	800
20000	15	24	1400	950
30000	19	36	2500	1200
40000	25	39	2800	1800
50000	35	45	3500	2200
60000	62	70	3200	2500
70000	70	90	4200	3000
80000	89	110	4500	3600
90000	95	120	4600	4500
100000	100	140	5200	4800

Tableau 5. 3: Expérimentations de L'approche K-means base BSO.

D'après ce tableau ci-dessus, on remarque que la performance de la première approche diminue à chaque fois qu'on augmente le nombre de données à regrouper, de façon lorsqu'on traite un nombre de données (1000 données) on aura (Fitness de la Meilleur solution=5, pour jump et 8 pour counter). Cependant quand le nombre de données atteint 100000, (Ftiness de la Meilleur solution=100 pour jump et 140 pour counter). Ainsi, on peut dire que le temps d'exécution de counter est meilleur que dans jump alors lorsqu'il s'agit de la fitness jump est meilleur que counter.

b. L'approche « BSO base K-means »

En fixant K à 80, IMAX à 100, stratégie de détermination de Search Area à Hybride. Et on varie le calcul de voisinage et la taille de données. On aura les résultats suivants :

Taille de benchmark	La meilleure fitness obtenue		Le Temps d'exécution en (SEC)	
	Jump	Counter	Jump	Counter
10000	8	12	1500	700
20000	18	20	1300	880
30000	20	22	2200	1000
40000	26	34	1800	1500
50000	30	40	2500	2000
60000	40	55	3000	2200
70000	52	60	4000	3500
80000	60	80	4100	3800
90000	78	95	4800	4200
100000	80	150	5100	4500

Tableau 5. 4: Expérimentations de L'approche BSO base K-means.

D'après ce tableau ci-dessus, on remarque que la performance de la deuxième approche diminue à chaque fois qu'on augmente le nombre de données à regrouper, de façon lorsqu'on traite un nombre de données (1000 données) on aura (Fitness de la Meilleur solution=8, pour jump et 12 pour counter).

Cependant quand le nombre de données atteint 100000, (Fitness de la Meilleur solution=80 pour jump et 150 pour counter). Ainsi, on peut dire que le temps d'exécution de counter est meilleur que dans jump alors lorsqu' il s'agit de la fitness jump est meilleur que counter.

Selon les résultats précédents nous remarquons que la performance diminue en augmentant le nombre de règles à fouiller. La première approche est meilleure lorsqu'il s'agit d'une petite taille de données. Cependant, lorsqu'il s'agit d'une taille de données importantes la deuxième approche peut être considérer. On Conclut que la deuxième approche est plus performante que la première approche.

4. Conclusion

Dans ce chapitre nous avons pu mettre en œuvre les deux approches de classification non supervisée (présentées dans le quatrième chapitre) sur des benchmarks construits aléatoirement varient de 10000 données jusqu'au 10000 données dans l'application **BSOCK** (**B**ees **S**warm **O**ptimization based on **C**lustering **K**-means). D'après les différentes expérimentations nous avons pu démontrer l'efficacité de ces approches dans la classification non supervisée, de telle sorte la fitness de la solution finale n'a jamais dépassé 150 en terme de distance.

De plus, on a réussi à montrer la performance de la deuxième approche par rapport à la première approche lorsqu'il s'agit de taille de données importantes en entrée de l'application.

Comme future et perspective de cette application, on essaye de tester l'application sur des benchmarks connus comme iris et glass. Et comparé les approches proposées avec d'autres approches existantes dans la littérature.

Conclusion Générale

Conclusion Générale

La question posée dans notre thèse est de savoir comment partitionner des objets de données dans un nombre déterminé de groupes. Nous considéronsle regroupement comme un problème d'optimisation combinatoire.

L'objectif principal de notre algorithme est d'obtenir un regroupement de haute qualité, avec une simplicité de mise en œuvre. Le k-means est la méthode la plus largement utilisée,probablement en raison de leur facilité en mise en œuvre et des résultats raisonnables. Les meilleurs regroupements sont obtenus à l'aide des techniques d'optimisation comme tabou et les algorithmes génétiques.

Nous introduisons un nouvel algorithme de clustering basé sur la colonie des abeilles. Les questions posées dans la conception sont la détermination des voisinages, et les stratégies de recherche.

La fonction de voisinage équilibre entre les régions, et apporte des changements au regroupement, et en même temps, converge vers un optimum local.

La stratégie de recherche s'intéresse de déterminer des grands champs de recherche pour trouver la solution minimale.

Dans notre travail on a fait une sorte d'hybridation entre k-means et BSO (BSOCK), tel qu'on a proposé trois stratégies afin d'explorer l'espace de recherche, et ces stratégies sont : Stratégie saut avec flip,Stratégie suivant avec flip est laStratégie hybride.On a aussi proposé deux stratégies pour déterminer les voisinages pour chaque régions de l'espace de recherche est ces stratégies sont:Stratégie jump et Stratégie counter.

Ensuite, on a proposé deux approches.Dans la première approche, on initialise les centres de gravité de K-means en utilisant la métaheuristique BSO. Après on s'exécute le K-means tout entière. Par contre, dans la deuxième approche, on applique K-means ensuite BSO.

Les résultats des deux approches sont très encourageants.De plus, on a pu résoudre le problème de K-means.

Bibliographie

Bibliographie

[1] www.loria.fr/devignes/M2P/FouilleDonnee-M2P-oc2006-Devignes. Consultation 2013.

[2] Djemia Nora *'Approche métaheuristique pour optimisation Multi objectif Discrète'*, Mémoire de magister, Université de Blida, (Novembre 2012).

[3] Stéphane TURRERY, *'Datamining et statistique décisionnelle l'intelligence des données'*, Universités de Rennes 1 et de Paris-Dauphine, 2007, p 44.

[4] http:\\morgon.univlyon2.fr\ (consultation: 2012).

[5] Philippe PREUX, *'Fouille de données'*, Notes de cours, Université de Lille 3, 31 août 2009.

[6] Nicolas Pasquier, *'Data mining clustering'*, France, p52.

[7] A.K. JAIN, M.N.MURTY, P.J. FLYNN, *'Data Clustering'* ACM Computing Surveys, Vol. 31, No. 3, Septembre 1999.p 323.

[8] E-G. Talbi, *Fouille de données (Data Mining),* University of Lille.

Bibliographie

[9] Mess Aoudiouchene et Mohamed Boumahdi Salima, '*Résolution du problème de partitionnement matériel/logiciel par l'aproche des colonies des fourmis'*, Mémoire d'ingénieur. Université de Blida, (2006).

[10] Laetitia Jourdan,'*Extraction de connaissances à l'aide des métaheuristique* ', thèse de doctorat, Lille, France,2003.

[11] Jiawei Han end MichelineKamber,'*Data Mining Concepts and Techniques'*,San Francisco, Diane Cerra, 2006, 772p.

[12] Michel GENDREAU *and* Jean-Yves POTVIN,' *Handbook of Metaheuristics'*, Montreal Canada, March 2010, 699p.

[13] F.Glover,' *Future paths for integer programming and links to artificial intelligence'*. Computer and operations research 13533-549,1986.

[14] Jin-Kao Hao,' Métaheuristiques et leurs applications ', Université d'Angers

[15] Baptiste AUTIN. ' *Les métaheuristiques en optimisation combinatoire'*. Mémoire d'obtenir l'examen probatoire en informatique. Conservatoire National Des Arts Et Métiers-Paris-, (Mai 2006).

[16] Sean Luke, '*Essentials of Métaheuristiques'*, Sean Luke, March, 2009, 233p.

Bibliographie

[17] Souquet Amédée, Radet Francois Gérard, *'Algorithme Génétique'* TE de fin d'année, 2004.

[18] www-igm.univ-mlv.fr.

[19] http://www.recherche.enac.fr.

[20] Marc Sevaux, ' *Mtaheuristiques Des outils pour l'optimisation et la robustesse,* Le Mont Houy et Bat Jonas, France, 2005,47p.

[21] El-Ghazali Talbi, *'Metaheuristics from design to implementation'*, Canada, John Wiley & Sons, 2009, 618p.

[22] www.i3s.unice.fr.(consultation 2013)

[23] Arnaud Liefooghe, Laetitia Jourdan,' *introduction aux métaheuristiques*', institut national de recherche en informatique et en automatique, Lille, Paris, juin 2008.

[24] Sbai ahmed, '*résolution de problème du voyageur de commerce par les algorithmes génétiques'*, université de Bida, 2011.

[25] reussirlem2info.files.wordpress.com. (Consulter 2013)

[26] K.Michael, Joyce C. Wong, *'Clustering categorical data sets using tabu search techniques*', Pattern Recognition 35 (2002) 2783 – 2790, Received15 March 2001; received in revised form 10 September 2001; accepted20 November 2001.

[27] P. Fra¨nti1 and J. Kivija¨rvi2, '*Randomised Local Search Algorithm for the Clustering Problem*', Pattern Analysis & Applications (2000)3:358–369.

[28] Bashar Al-Shboul and Sung-Hyon Myaeng, "*Initializing K-Means using Genetic Algorithms*", World Academy of Science, Engineering and Technology 54 2009.

[29] Dharmendra K Roy and Lokesh K Sharma, "*Genetic K-Means Algorithm for Mixed Numeric And Categorical Data Sets*", International Journal of Artificial Intelligence & Applications (IJAIA), Vol.1, NO.2, April 2010.

[30] Wang Min and Yin Siqing, "*Improved K-means Clustering Based on Genetic Algorithm*", 20IO International Conference on Computer Application and System Modeling (ICCASM 2010)

[31] R.J. Kuo, L.M. Lin, « *Application of a hybrid of genetic algorithm and particle swarm optimization algorithm for order clustering* », Decision Support Systems 49 (2010) 451–462.

[32] Arpit Jain, Shikha Agrawal, Jitendra Agrawal and Sanjeev Sharma "*Analysis of Population Based Metaheuristic Used for Gene Clustering*", *International Journal of Computer and Communication Engineering, Vol. 2, No. 2, March 2013.*

[33] Ziainia Mourad et Ayad Omar, '*Application de la métaheuristique des colonies de fourmis pour le problème de repliement de protéine'*, Mémoire d'ingénieur d'état, Université de Blida, 2008.

[34] Touari Chafia, Taguida anissa,'*optimisation par colonie de fourmi pour la planification automatique des livraisons'*, Mémoire d'ingénieur d'état, Université de Blida, 2008.