



Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Mohamed El-Bachir
El-Ibrahimi
Bordj Bou Arréridj
Faculté des mathématiques et de
l'informatique
Département de Recherche
Opérationnelle



Mémoire de Master

Mémoire présenté en vue de l'obtention du diplôme de Master en :

Domaine : Mathématiques et Informatique.

Filière : Mathématiques.

Spécialité : Méthodes et outils pour la recherche opérationnelle.

*solving multiobjectif portfolio optimization
problems using firefly algorithm*

Présenté par :

- DEGHACHE Selma
- OUCHENE Hadjer

Sous la direction de :
M. ZOUACHE DJAFER

Soutenu publiquement le Jour/Moi/année devant le jury composé de :

M. MAZA Sofiane	Université de BBA	Président
M. ZOUACHE Djafer	Université de BBA	Encadreur
M. ALLOU Lotfi	Université de BBA	Examinateur

Année Universitaire 2021/2022



Remerciements

*Nous tenons à remercier tout premièrement Dieu le tout puissant pour la volonté la santé et la patience, qu'il nous donné durant toutes ces longues années. Ainsi, nous tenons également à exprimer nos vifs remerciements à notre encadreur **Mr. ZOUACHE Djafer** pour avoir d'abord proposée ce thème, Pour suivi continuel tout le long de la réalisation de ce mémoire et qui n'a pas cessé de nous donner ses conseils. Nous tenons à remercier vivement toutes personnes qui nous ont aidés à élaborer et réaliser ce mémoire, ainsi à tous ceux qui nous ont aidés de près ou de loin à accomplir ce travail. Nos remerciements vont aussi à tous les enseignants qui ont contribué à notre formation par ailleurs, Nos remerciements à tous les membres du jury qui ont accepté de juger notre travail. En fin, nous tenons à exprimer notre reconnaissance à tous nos amis et collègues pour le soutien moral et matériel...*

*D*édicace

Je dédie ce modeste travail à :

*Mon frère, « **ouchene Ayoub** » que dieu lui fasse miséricorde.*

A mon père ouchene :

e qui a sacrifié sa vie afin de me voir grandir et réussir dans le parcours de l'enseignement. Celui qui a toujours resté à mes côtés dans les moments pénibles de ma vie.

A ma très chère mère benkechida :

qui m'a accompagné durant les moments les plus rudes de ce long parcours de mon éducation, celle qu'a fait preuve de ces plus nombreux projets pour me permettre de goûter le fardeau de ce monde et de chercher la voie de ma vie avec ces précieux conseils, dont je devais incessamment être de grande compétence et motivation. Cependant. Je prie Dieu le Miséricordieux qu'il te portera récompense, car la mienne ne sera guère complète.

*Une spéciale dédicace pour tous mes oncles surtout **Miloud**.*

A mes sœurs et mon frère « kadafi ».

A Ma Famille et mes proches.

A mon binôme D. Selma.

A Tous mes amis.

A vous.

Ouchene Hadjer

Je dédie ce succès à:

*Un morceau de mon cœur, mon père **DEGHACHE Chaaban***

que Dieu lui fasse miséricorde.

*La femme qui m'a bien élevé et qui a veillé tout la nuit pour moi, c'est ma mère **ZEHAR Souhila**, tu es joie de vivre à mes yeux.*

A ma grand-mère qui m'a conseillé de suivre le chemin du succès, qui Dieu lui fasse miséricorde.

*A mon grand-père **ZEHAR Ammar** qui avait un deuxième père j'en ai marre d'être ici*

*A ma petite et unique sœur **DECHACHE Djamila**.*

A toute ma famille

A tous mes professeurs qui m'ont enseigné durant mon parcours Universitaire.

*A mon binôme **OUCHENE Hadjer***

A tous mes amis

A vous

DEGHACHE Selma

Table des matières

1	Optimisation combinatoire	9
1.1	Introduction	9
1.2	Problème d'optimisation combinatoire	9
1.2.1	Définition	9
1.2.2	Notion de base	10
1.2.3	Résolution d'un problème d'optimisation combinatoire	11
1.2.4	Complexité Algorithmique	11
1.2.4.1	Définition de la complexité	11
1.2.4.2	Classes de complexité	12
1.3	Problème d'optimisation de portefeuille	13
1.3.1	Éléments d'optimisation de portefeuille	14
1.3.1.1	Un actif financier	14
1.3.1.2	Un portefeuille	14
1.3.1.3	Le rendement	14
1.3.1.4	Analyse du risque	14
1.3.2	La théorie moderne du portefeuille	15
1.3.3	Modèles mathématiques pour l'optimisation de portefeuille	16
1.3.3.1	Les objectifs :	16
1.3.3.2	Les contraintes	17
1.3.3.3	Le modèle standard de Markowitz	19
1.3.3.4	Modèle d'optimisation de portefeuille multi-objectif	19
1.4	Méthode de résolution	20
1.4.1	Les méthodes exactes (complètes)	21
1.4.1.1	La programmation dynamique	21
1.4.1.2	La méthode Branch and bound (B&B)	22
1.4.2	Les méthodes approchées (incomplètes)	23
1.4.2.1	Les méthodes heuristique	23
1.5	Conclusion	24
2	Les métaheuristiques	25
2.1	Introduction	25
2.2	Les méta-heuristiques à base de solution unique	25
2.2.1	La recherche locale simple (la descente)	26

2.2.2	Le recuit simulé	26
2.2.3	La recherche à voisinage variable	29
2.3	Les méta-heuristiques à base de population de solutions	29
2.3.1	Les algorithmes évolutionnaires	30
2.3.2	L'algorithme génétique	32
2.3.2.1	Le codage des individus	35
2.3.2.2	La sélection	37
2.3.2.3	Le croisement	37
2.3.2.4	La mutation	38
2.3.2.5	Le remplacement	39
2.3.2.6	La recherche par harmonies	39
2.4	les algorithmes à base de l'intelligence par essaim	40
2.4.1	L'intelligence par essaim de particules	40
2.4.2	Algorithme FireFly	42
2.4.2.1	Inspiration	42
2.4.2.2	Les lucioles naturelles	42
2.4.2.3	Définition (Algorithme firefly)	43
2.4.2.4	Caractéristiques de l'algorithme Firefly	43
2.5	Conclusion	45
3	Optimisation multi-objectif	47
3.1	Introduction	47
3.2	Problème d'optimisation multi objectif	47
3.2.1	Exemple	48
3.2.2	Notion de base et quelque définitions	49
3.2.2.1	dominance au sens Pareto	49
3.2.2.2	front Pareto	49
3.2.2.3	convexité	50
3.2.2.4	le point idéal	51
3.2.2.5	le point Nadir	51
3.2.2.6	espace de décision	51
3.2.2.7	espace objectif de dimension	51
3.3	Approches de résolution multi objectif	51
3.3.1	La recherche des solutions de meilleur compromis	52
3.3.2	Le choix de la solution à retenir	52
3.3.2.1	Classification « point de vue décideur »	52
3.3.2.2	Classification « point de vue concepteur »	53
3.4	Approches non Pareto	53
3.4.1	Les approches scalaires « ces approches sont de type a priori »	54
3.4.1.1	Approche d'agrégation	54
3.4.1.2	But programmé	55
3.4.1.3	Approches ϵ -contraintes	55
3.4.1.4	Conclusion sur Les approches scalaires	55
3.4.2	Les approches non scalaires non Pareto « ces approches sont de type a posteriori »	56
3.4.2.1	Sélection parallèle	56
3.4.2.2	Sélection lexicographique	56
3.4.2.3	Conclusion sur Les approches non scalaires	56

3.5	Approches Pareto	57
3.6	Conclusion	57
4	FA pour la sélection de portefeuille	58
4.1	Introduction	58
4.2	sélection de portefeuille	58
4.3	Représentation et encodage de la solution	59
4.4	FireFly algorithm pour PSP	59
4.4.1	Initialisation	60
4.4.2	Évaluation	61
4.4.3	L'attractivité de FireFly	62
4.4.4	Attractiveness calculation	62
4.4.5	Le mouvement towards attractive fire fly	62
4.5	Réalisation	62
4.5.1	Description du langage	62
4.5.1.1	Présentation des résultat	63
4.5.1.2	Paramètres	63
4.5.1.3	Source des données	64
4.6	Conclusion	69
	Bibliographie	71

Table des figures

1.1	Quelques classes de complexité	12
1.2	Frontière des portefeuilles efficaces selon Markowitz	16
1.3	Taxonomie des méthodes de résolution	21
1.4	Divisé en sous-problèmes	22
1.5	Méthode de séparation/évaluation.	23
1.6	Exemple d'algorithme glouton pour répondre au problème du voyageur de commerce.	24
2.1	Un schéma d'évolution d'une recherche locale simple [19]	26
2.2	Organigramme d'un algorithme évolutionnaire	31
2.3	Les types des algorithmes évolutionnaires.	32
2.4	Démarche d'un algorithme génétique. [32]	33
2.5	Codage arborescent d'un chromosome	37
2.6	Organigramme de l'algorithme d'optimisation par essaim de particules.	41
2.7	Une luciole naturelle	43
2.8	Une vue conceptuelle des relations de l'algorithme FireFly	45
3.1	Espace de décision et espace objectif d'un problème d'optimisation multi-objectif.	48
3.2	Relation de dominance (Cas de deux objectifs à maximiser)	49
3.3	Front de Pareto pour un espace à deux dimensions	50
3.4	espace convexe à (gauche), non convexe à (droite)	50
3.5	Points caractéristiques d'un problème de maximisation bi objectif	51
3.6	Classification « point de vue concepteur »	53
3.7	l'approche d'agrégation	54
4.1	Meilleure solution avec des coefficients λ différents à chaque génération	63
4.2	Le rendement espéré et la variance de chaque actif	64
4.3	La matrice de Corrélation	65
4.4	La matrice de covariance	68

Liste des Algorithmes

1	La recherche locale simple (la descente)	27
2	Le recuit simulé	28
3	la recherche à voisinage variable	30
4	L'algorithme génétique	35
5	L'algorithme général de la recherche par harmonies	40
6	l'algorithme Firefly	44
7	FireFly pour résoudre les PSP	60

Liste des tableaux

2.1	Codage binaire d'un chromosome	36
2.2	Codage réel d'un chromosome	36
2.3	Codage entier d'un chromosome	36
2.4	Codage à caractère d'un chromosome	36
2.5	Croisement 1-point.	38
2.6	Croisement 2-point.	38
2.7	Croisement uniforme.	38
4.1	un exemple de décodage	59

Introduction générale

La recherche opérationnelle appelée aussi aide à la décision peut être définie comme l'ensemble des méthodes et techniques rationnelles d'analyse et de synthèse des phénomènes d'organisation utilisables pour élaborer de meilleures décisions.

L'optimisation combinatoire consiste à trouver parmi un ensemble discret de solutions respectant des contraintes une solution qui optimise une fonction objectif, Par optimiser, on entend trouver la plus petite valeur (problème de minimisation) ou la plus grande valeur (problème de maximisation).

De nombreux problèmes peuvent se modéliser comme des problèmes d'optimisation combinatoire. De manière large, les problèmes d'optimisation combinatoire peuvent se ranger dans deux classes de complexité : la classe P et la classe NP-difficile. Dans la première classe, on trouve les problèmes considérés comme "faciles", c'est-à-dire pour lesquels il existe un algorithme polynômial pour trouver la solution. On ne connaît pas de tels algorithmes pour les problèmes de la seconde classe. Dans ce document, on s'intéresse principalement à des problèmes NP-difficiles.

Problématique :

En effet, la gestion du portefeuille consiste à investir des actifs dans le but d'accroître la valeur du patrimoine financier ou de réaliser certains projets Puisqu'elle peut se faire sur différents horizons de placements et doit prendre en considération vos besoins de liquidité et les impacts fiscaux de vos décisions, elle requiert expertise, expérience et objectivité. Les placements doivent être diversifiés par secteurs, catégories d'actif, pays et styles de gestion et rééquilibrés dans le temps pour réduire le risque lié aux rendements des marchés, tout en s'adaptant aux changements. En d'autres mots, la théorie du portefeuille doit viser l'appréciation de votre capital tout en minimisant la volatilité de votre portefeuille.

L'objectif :

Dans ce travail, on fait une étude sur la théorie du portefeuille, en essayant de trouver une manière dont on peut construire un portefeuille qui maximise le rendement avec un risque contrôlé, ce qui revient à minimiser le risque avec un rendement minimum fixé et le résoudre en utilisant l'algorithme de Firefly puis de comparer les résultats obtenues.

Organisation du mémoire :

Afin de bien présenter notre travail nous l'avons structuré de la manière suivante :

Chapitre 01 : Aborde la présentation générale de l'optimisation combinatoire

Chapitre 02 : Porte sur les méta heuristiques .

Chapitre 03 : Aborde la présentation générale de l'optimisation multi objectif

Chapitre 04 : Dans le quatrième chapitre, nous donnons l'implémentation et les résultats numériques de la programmation de portefeuille par l'algorithme FireFly.

Optimisation combinatoire

1.1 Introduction

L'optimisation combinatoire occupe une place très importante en recherche opérationnelle, en mathématiques discrètes et en informatique. Elle consiste à comprendre un problème réel et pouvoir le transformer en un modèle mathématique à fin d'extraire ses propriétés structurelles et le résoudre. Bien que les problèmes d'optimisation combinatoire soient souvent faciles à définir, ils sont généralement difficiles à résoudre.

Nous allons discuter dans ce chapitre des différents problèmes d'optimisation ainsi que de leurs méthodes de résolution. Ce chapitre sera organisé comme suit : Nous commençons par l'étude de problèmes d'optimisation combinatoire dans la section 1, en suite nous définie le problème d'optimisation de portefeuille dans la section 2. la section 3 en a expliquer quelque méthode de résolution (exacte et approcher), en fin la section 4 conclut ce chapitre

1.2 Problème d'optimisation combinatoire

1.2.1 Définition

L'optimisation est une discipline mathématique qui, bien qu'omniprésente depuis des lustres, a pleinement pris son essor au cours du $x x^e$ siècle d'une part, sous la stimulation du développement des sciences de l'industrie et de la planification, telles que l'économie, la gestion, etc. . . , et des sciences appliquées aux technologies naissantes, comme l'automatique, le traitement du signal, etc. . . , et d'autre part grâce au développement de l'informatique qui a rendu efficiente ses méthodes algorithmiques jusque là impraticables.[1]

Optimiser c'est choisir parmi plusieurs possibilités celle qui répond le mieux à certains critères. L'optimisation, et plus généralement la recherche opérationnelle, intervient dès lors pour appliquer l'outil mathématique à cette résolution, si tant est que le problème soit formalisable mathématiquement. De nos jours son champ d'application est on ne peut plus vaste : optimisation des ressources, des gains, des coûts dans l'industrie, optimisation du trafic aérien, ferroviaire, routier, dans le transport, optimisation de la couverture radar, de la réactivité d'intervention, de la gestion des stocks et des troupes dans le domaine

militaire, etc. . . , sans parler des sciences dures, physique, chimie, informatique, automatique, traitement du signal, etc. . . , pour lesquels nombre de problèmes se ramènent et se résolvent par optimisation. C'est une discipline fondamentale dans les sciences de l'ingénierie, de l'économie et de la gestion, pour ne citer qu'elles[2]

$$\begin{cases} \min_x f(x) \\ \text{s.c } g_i(x) \leq 0, i = 1, \dots, m \\ h_j(x) = 0, j = 1, \dots, p \\ x \in S \end{cases} \quad (1.1)$$

1.2.2 Notion de base

Tout d'abord, nous définissons les notions communes à n'importe quelle méthode d'optimisation :

Définition 1.1 (Vecteur de décision) *Un vecteur de décision est un vecteur correspondant à l'ensemble des variables du problème, il est noté : $X^* = [x_1, x_2, \dots, x_n]^T$ avec : n le nombre de variables ou dimension du problème x_k et la variable sur la dimension k*

Définition 1.2 (Fonction objectif) *est un critère sur lequel sont jugés les vecteurs de décision pour déterminer le meilleur vecteur. Un critère peut être une variable du problème ou une combinaison de variables.*

Définition 1.3 (Contraintes) *Une contrainte du problème est une condition que doivent respecter les vecteurs de décision du problème. Une contrainte est notée : $g_i(x^*)$ avec $i = 1, \dots, m$ est le nombre des contraintes.*

Définition 1.4 (Minimum local) *Un point $x \in X$ est un minimum local du problème si et seulement si : $\forall x' \in \bigvee(x), f(x) \leq f(x')$ ou $\bigvee(x)$ définit un voisinage de x .*

Définition 1.5 (Minimum global) *Un point $x \in X$ est un minimum global du problème si et seulement si : $\forall x' \in X, f(x) \leq f(x')$ (il appartient alors à l'ensemble des minimisera)[3]*

Définition 1.6 (solution réalisable) *on dit que X est une Solution réalisable sil vérifiant les contraintes du problème. $X = \{x \in R_n : Ax \leq b, x \geq 0\}$*

Définition 1.7 *Un sous-ensemble C de R_n est dit convexe si $\forall x, y \in C, \forall \lambda \in [0, 1], \lambda x + (1 - \lambda)y \in C$.*

Définition 1.8 *On dit qu'une fonction f est convexe sur un intervalle I si et seulement si :*

$$\forall (x_1, x_2) \in I^2; \forall \lambda \in [0, 1], f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2) \quad (1.2)$$

Remarque 1 *On dit qu'une fonction f est concave sur un intervalle I si et seulement si f est convexe.*

1.2.3 Résolution d'un problème d'optimisation combinatoire

La résoudre au problème d'optimisation combinatoire nécessite l'étude de trois points particuliers :

- La définition de l'ensemble des solutions réalisables,
- L'expression de l'objectif à optimiser,
- Le choix de la méthode d'optimisation à utiliser,

Les deux premiers points relèvent de la modélisation du problème, le troisième de sa résolution. Afin de définir l'ensemble des solutions réalisables, il est nécessaire d'exprimer l'ensemble des contraintes du problème. Ceci ne peut être fait qu'avec une bonne connaissance du problème sous étude et de son domaine d'application.

1.2.4 Complexité Algorithmique

Dans la théorie de la complexité, on mesure la complexité d'un problème, par le nombre de ressources nécessaires (temps, espace mémoire, etc.) à sa résolution par le meilleur algorithme connu. Il s'agit donc d'étudier la difficulté des problèmes, de les organiser par classes de complexité et d'étudier les relations entre les classes de complexité.

1.2.4.1 Définition de la complexité

On distingue deux types de problèmes :

- **Problèmes de décision** En prenant l'exemple du voyageur de commerce, il s'agit de déterminer s'il existe un circuit qui passe une et une seule fois par toutes les villes de longueur inférieure à k , étant donné un ensemble de villes, les distances entre villes et un entier k . La réponse à un problème de décision est soit oui soit non. Ainsi, pour ce problème, on est amené à répondre à cette question par affirmation s'il existe un tel chemin.
- **Problèmes d'existence de solutions** La question ici est de la forme « trouver un élément tel que... » et sa réponse consiste à fournir un tel élément; dans le cas du problème du voyageur de commerce, ça revient à trouver un circuit de longueur inférieure à k . On peut toujours trouver à partir des problèmes de recherche de solution, un problème décidable équivalent.

En complexité, on s'intéresse beaucoup plus aux premiers types de problèmes.

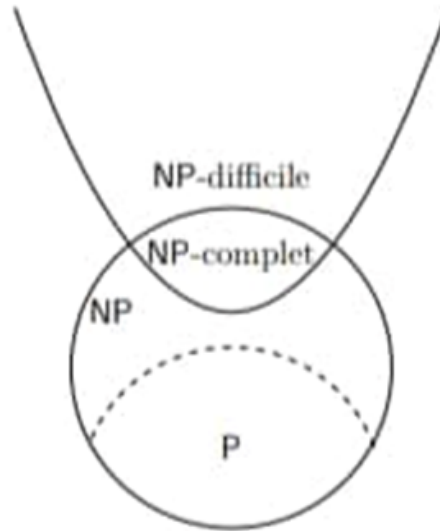


FIGURE 1.1 – Quelques classes de complexité

Instances d'un problème

On parle d'instances du problème pour désigner les données de ce problème. Une instance du problème du voyageur de commerce est donc la donnée de villes et de distances séparant les villes. Comme on mesure la complexité en fonction de la taille d'une instance, la représentation (le codage) d'une instance joue un rôle important.

1.2.4.2 Classes de complexité

Nous allons détailler ici quelques classes de complexité jugées utiles pour les problèmes traités dans ce manuscrit. La figure 1.1 montre ces classes sous l'hypothèse $P \neq NP$. [4].

classe P :

La classe de complexité **P** est l'ensemble de tous les problèmes de décision qui peuvent être résolus au pire des cas dans un temps polynomial. En d'autres termes, un problème est dans **P**, s'il est un problème de décision et s'il existe un algorithme qui résout n'importe quelle instance de taille n du problème en un temps, au plus égal à $O(n^k)$, pour un entier k . (n serait le nombre de bits d'un encodage raisonnable de la solution). La classe **P** serait donc la classe des problèmes décisionnels auxquels nous avons des algorithmes en temps polynomial. [5]

Un exemple d'un problème dans la classe **P** est la multiplication de deux nombres de n chiffres. L'algorithme le plus évident pour cette opération résout ce problème en un temps égal à $O(n^2)$.

classe NP :

La classe **NP** est celle des problèmes dont les solutions peuvent être vérifiées en temps polynomial. Les initiales N et P de cette classe veulent dire Non déterministe Polynomial. Ces problèmes peuvent être résolus en temps polynomial par une machine de Turing non-déterministe.

Par exemple, factoriser un nombre est un problème **NP**. Donner une factorisation d'un grand nombre prend un temps considérable, par contre, vérifier une factorisation donnée

d'un nombre est une tâche largement plus facile à réaliser.

On sait que $P \subset NP$ mais on ne sait pas si ces deux classes sont égales i.e. $P = NP$. La majorité des scientifiques croient que $P \neq NP$, et cette question reste toujours ouverte.

NP-difficile Un problème NP-difficile est au moins aussi difficile que n'importe quelle autre problème dans NP. Un problème A est NP-difficile, si pour chaque problème L dans NP, il existe une réduction en temps polynomial de L en A.

Les problèmes NP-difficile ne sont pas nécessairement des problèmes de décidabilité et ils n'appartiennent pas forcément à la classe NP.

NP-complet Un problème NP-complet est un problème NP et il est NP-difficile en même temps. Chaque problème NP peut être réduit ou transformé (en un temps polynomial) en un problème NP-complet. Cette classe de problème est un peu particulière car résoudre n'importe quel problème NP-complet, résoudra tous les problèmes de la classe NP.[6]

Beaucoup de problèmes sont NP-complet, citons par exemple :

- Problème du sac à dos.
- Coloriage de graphes.
- Satisfiabilité booléenne

Nous allons montrer dans la section suivante le problème d'optimisation de portefeuille

1.3 Problème d'optimisation de portefeuille

Définition 1.9 *Un portefeuille est celui qui présente le couple risque/rentabilité optimal pour un investisseur.*

Développée par l'économiste américain Harry Markowitz (l'un des Prix Nobel d'économie 1990), la théorie du portefeuille[7] définit le mécanisme de sélection de titres constituant un portefeuille boursier idéal sur le papier.

Selon H. Markowitz, un portefeuille est efficient si :

- *Il permet d'atteindre un niveau de rendement anticipé fixé avec le minimum de risque (c'est-à-dire le minimum de variance); ou*
- *Il fournit, pour un niveau de risque donné, la possibilité d'atteindre le maximum de rendement moyen escompté.*
- *H. Markowitz distingue deux portefeuilles : un portefeuille efficient purement risqué et un portefeuille efficient non risqué.*

Dans les deux cas, il s'agit de trouver la combinaison optimale de X titres risqués (ou non risqués) grâce auxquels un investisseur obtiendra le niveau maximal de rendement espéré pour un niveau de variance (risque) donné.

1.3.1 Éléments d'optimisation de portefeuille

1.3.1.1 Un actif financier

C'est un titre ou un contrat produisant à son propriétaire des revenus ou un gain en capital sur le marché financier. Cet actif produit une certaine prise de risques et peut être transmis ou négocié sur le marché. Il est considéré comme un placement et est comptabilisé dans le patrimoine de l'individu.

1.3.1.2 Un portefeuille

C'est un ensemble d'actifs financiers détenus par un investisseur. Ces actifs peuvent provenir de différentes classes : actions, obligations, produits dérivés, matières premières, fonds, cash, etc. L'investisseur, pour diminuer son risque, procède souvent à une diversification de ses actifs. Ces derniers possèdent chacun une volatilité qui leur est propre et sont plus ou moins corrélés entre eux. La détention de plusieurs actifs différents tend donc, généralement, à diminuer la volatilité globale du portefeuille. Le degré de risque (nature des titres détenus, volatilité historique, diversification), le dynamisme de la stratégie (fréquence de réajustement de l'allocation) et le rendement obtenu sont des caractéristiques importantes, servant à comparer les portefeuilles boursiers entre eux.

1.3.1.3 Le rendement

Le rendement d'un actif est une variable aléatoire et, le rendement d'un portefeuille est une combinaison linéaire pondérée des actifs qui le composent. Par conséquent, le rendement d'un portefeuille est également une variable aléatoire et possède une espérance et une variance.

1.3.1.4 Analyse du risque

L'utilisation d'outils de mesure du risque est devenu systématique et les professionnels ont développé des instruments très sophistiqués. Néanmoins, il existe bon nombre d'outils constituant la base de la gestion du risque, à la portée de tous les investisseurs et ayant démontré leur efficacité. Nous abordons ici les plus célèbres et utilisés d'entre eux :

Définition 1.10 (Covariance et la variance) *La variance et la covariance sont deux des principales mesures utilisées pour mener à bien une étude.*

Ces deux instruments sont considérés comme la base de toute étude de risque qu'il faut connaître à tout prix avant même d'entreprendre de manipuler un portefeuille.

Définition 1.11 (Variance) *Selon la définition classique, la moyenne des carrés des écarts par rapport à la moyenne. En termes plus mathématiques elle peut être considérée comme une mesure servant à caractériser la dispersion d'une distribution ou d'un échantillon.*

$$V(X) = E((X - E(X))^2) = \frac{\sum_{i=1}^n (x_i - \tilde{X})^2}{n} \quad (1.3)$$

avec :

- x_i : le cours de l'actif x à l'instant i ;
- \tilde{X} : moyenne du cours de l'actif x ;

- n : nombre de périodes.

Grossièrement on peut la voir comme la moyenne des carrés moins le carré des moyennes. Cette formule intègre des carrés dans le but d'éviter que les écarts positifs et les écarts négatifs par rapport à la moyenne ne s'annulent.

La dimension de cette mesure étant le carré de la dimension de la moyenne, on utilise plus souvent l'écart-type qui n'est rien d'autre que la racine de la variance.

Définition 1.12 (Covariance) La covariance est légèrement différente. Si la variance permet d'étudier les variations d'une variable par rapport à elle-même, la covariance va permettre d'étudier les variations simultanées de deux variables par rapport à leur moyenne respective.

La covariance peut être vue comme le produit des valeurs de deux variables moins le produit des deux moyennes. Mathématiquement, la formule est la suivante :

$$cov(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \tilde{X})(y_i - \tilde{Y}) \quad (1.4)$$

avec :

- x_i : cours de l'actif x à l'instant i ,
- y_i : cours de l'actif y à l'instant i ,
- \tilde{X} : moyenne du cours de l'actif x ,
- \tilde{Y} : moyenne du cours de l'actif y ,
- n : nombre de périodes.

Du résultat obtenu par cette mesure on en déduit que plus la covariance est faible et plus les séries sont indépendantes et inversement plus elle est élevée et plus les séries sont liées.

Une covariance nulle correspondant à deux variables totalement indépendantes.

1.3.2 La théorie moderne du portefeuille

La théorie moderne de portefeuilles (TMP), appelée aussi analyse moyenne-variance, est un modèle mathématique pour l'assemblage d'un portefeuille d'actifs, de sorte que le rendement espéré est maximisé pour un niveau de risque donné, défini comme variance. Son principe est que le risque et le rendement d'un actif sont évalués par la façon dont il contribue au risque et au rendement globaux d'un portefeuille [8], de tel sorte à trouver les portefeuilles dits " efficaces " qui maximisent le rendement grâce à la diversification en minimisant la corrélation entre les actifs.

Cette théorie est due à H. Markowitz, qui est appelée le père fondateur de cette théorie, après son travail intitulé "Portfolio sélection paru en 1952, pour lequel il a reçu un prix Nobel d'économie en 1990. Markowitz a établi la relation entre la moyenne et l'écart de l'investissement dans le cadre du compromis risque retour. Depuis lors, divers modèles élargis et améliorés ont été développés dans plusieurs directions. L'objectif du modèle de Markowitz vise à construire un portefeuille qui optimise simultanément le rendement et le risque, qui tient compte de l'incertitude liée au comportement des marchés financiers.

Cet objectif élimine un certain nombre des portefeuilles, moins efficaces que d'autres. La courbe qui relie l'ensemble des portefeuilles efficaces s'appelle la frontière efficace, dont la définition est donnée comme suit :

Définition 1.13 *L'ensemble efficace de Markowitz, aussi appelé frontière efficace, est un concept mathématique qui reflète les combinaisons ou les portefeuilles qui génèrent le rendement maximal prévu pour divers niveaux de risque.*

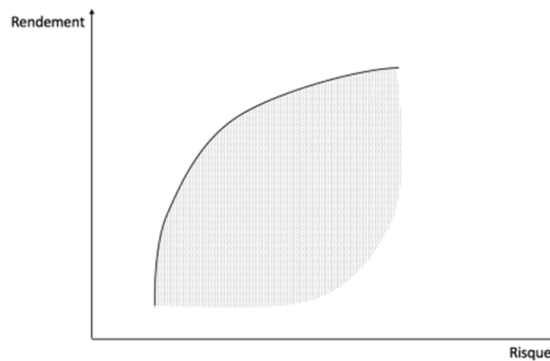


FIGURE 1.2 – Frontière des portefeuilles efficaces selon Markowitz

En dessous de cette courbe, tous les portefeuilles rejetés sont dits dominés.

1.3.3 Modèles mathématiques pour l'optimisation de portefeuille

Dans les années 1950, les premiers modèles de choix de portefeuille, comme l'approche moyenne-variance de Markowitz (1952), ont été élaborés dans un cadre discret mono périodique. La théorie de la variance moyenne (MV) de Markowitz apporte une solution standard au problème de la sélection des portefeuilles. Le risque d'un portefeuille peut être réduit en combinant des actifs avec des rendements imparfaitement corrélés. Après les travaux de Markowitz, plusieurs études ont tenté d'améliorer et d'étendre le modèle MV standard dans trois directions :

simplification du type et de la quantité de données d'entrée, application des autres mesures des risques et intégration des contraintes réelles.

Nos travaux portent sur cette dernière orientation et seront détaillés dans les prochaines sections. Plusieurs critères et contraintes différencient les modèles les uns des autres.

En effet, pour avoir un portefeuille en conditions réelles, il faut tenir compte de nombreuses contraintes, telles que la segmentation, la cardinalité, la quantité, la pré affectation, le lot rond, la logique et le budget

Ce chapitre décrit la plupart des modèles que l'on peut trouver dans la littérature. Il s'agit maintenant d'obtenir des proportions x_i d'actif dans le but de former un portefeuille qui puisse satisfaire le décideur. Nous commençons par décrire la majorité des objectifs rencontrés :

1.3.3.1 Les objectifs :

Depuis l'avènement théorique de la variance moyenne, l'optimisation du portefeuille dans les années 1950, il y a eu un débat continu sur la nécessité d'inclure dans le processus

des distributions de moments de rendement plus élevés : le coefficient d'asymétrie (skewness) et le coefficient d'aplatissement (kurtosis). Ces dernières années, l'importance croissante accordée par la réglementation aux mesures à la baisse du " risque de perte ", telles que la VaR et la CVaR, a accru l'intérêt pour ce sujet. Malheureusement, une grande partie de la documentation sur cette question ne tient pas compte de certains des éléments réels importants du problème, comme les coûts de transaction et les erreurs d'estimation des paramètres. Les objectifs liés à l'investissement varient selon les modèles considérés. Markowitz a présenté le modèle le plus représentatif de cette problématique en 1952 dans . Markowitz stipule que les objectifs qui seront considérés lors de l'optimisation d'un portefeuille se résument au rendement et au risque des historiques des actifs.

Le rendement :

Le rendement attendu sur un investissement financier est la valeur attendue de son bénéfice. Elle est obtenue en sommant les rendements de tous les scénarios possibles pondérés par la probabilité de réalisation de ses derniers.

$$\sum_{i=1}^n r_i x_i = 1$$

r_i : est le retour du scénario i ,

p_i : est la probabilité d'avoir le retour r_i dans le scénario i ,

n : est le nombre total de scénarios.

1.3.3.2 Les contraintes

Le risque :

Le risque de portefeuille est la probabilité que la composition de l'actif ou des parts des placements détenus ne permette pas d'atteindre les objectifs fixés sur le plan financier. Chaque placement au sein d'un portefeuille comporte ses propres risques, et un rendement potentiel plus élevé signifie généralement un risque plus élevé. En théorie, le risque de portefeuille peut être éliminé par une diversification adéquate : détenir des combinaisons de placements qui ne dépendent pas des mêmes conjonctures pour réaliser un bénéfice. En pratique, il est néanmoins plus probable que les risques soient minimisés et non pas totalement éliminés.

Le risque de portefeuille n'est qu'un des risques dont les investisseurs devraient se méfier. La plupart des risques s'appliquent à des placements individuels, mais il est également important de s'assurer que l'ensemble de la composition du portefeuille ne se répercute pas négativement sur les investisseurs (perte du capital). Mathématiquement, ce risque est calculé comme la variance du portefeuille, qui est une mesure de la fluctuation dans le temps du rendement réel global d'un ensemble de titres composant un portefeuille. Cette statistique sur les écarts de portefeuille est calculée en utilisant les écarts-types de chaque titre du portefeuille, ainsi que les corrélations de chaque paire de titres du portefeuille

La contrainte de budget :

Cette contrainte est la plus importante pour un investisseur. Il s'agit de s'assurer que la totalité du budget est répartie sur l'ensemble des actifs sélectionnés, en d'autres termes, la

somme des parts investies dans les différents actifs est égale à 100% du capital. Cette contrainte est représentée par l'équation :

$$\sum_{i=1}^n x_i = 1 \quad (1.5)$$

où :

n est le nombre d'actifs proposés pour l'investisseur.

Les contraintes de quantité :

Il existe certains coûts administratifs excessifs pour les très petites exploitations qui ont un rendement négligeable du portefeuille. Par conséquent, il est préférable d'éviter d'investir de petites quantités pour éviter de telles situations. D'autre part, il existe quelques politiques institutionnelles qui font qu'on ne puisse pas investir au-delà d'une certaine quantité sur un actif.

Exemple 1 *la politique de 51-49 établie en Algérie, où une entreprise étrangère ne peut détenir plus de 49% du capital d'une entreprise algérienne. Ces nouvelles limitations sont dites 'contraintes de quantités', ce sont des bornes inférieures et supérieures sur la proportion de chaque actif qui peut être détenue dans un portefeuille. L'équation représente la formulation mathématique de telles contraintes :*

$$\lambda_i y_i \leq x_i \leq \beta_i y_i \text{ avec } i = 1, \dots, n$$

où :

α_i et β_i représentent respectivement la borne inférieure et la borne supérieure qu'on puisse investir dans le i -ème actif.

Ces limitations s'appliquent uniquement sur les actifs sélectionnés, d'où nous multiplions par y_i qui est la variable de décision décrite dans la section

Contrainte de classe (Classe constraint) :

Certains investisseurs font le souhait de répertorier leurs actifs qui s'offrent à eux en plusieurs ensembles mutuellement exclusifs (classes). Chaque ensemble sera groupé avec des caractéristiques communes telles que les actifs des produits de santé, d'énergie, de consommation, etc. Ainsi les investisseurs choisissent au moins un actif de chaque catégorie pour constituer un portefeuille bien diversifié . La formulation mathématique d'une telle contrainte est la suivante

$$C_i \cap C_j = \phi \quad \forall i \neq j$$

Où C_m , $m = 1, \dots, M$ sont M ensembles disjoints de classes d'actifs.

La contrainte des classes exige qu'au moins un actif de chaque catégorie soit investi dans un portefeuille, et peut être définie comme suit :

$$S_i \in C_m, \quad m = 1, \dots, M.$$

1.3.3.3 Le modèle standard de Markowitz

La formulation standard du modèle de Markowitz, appelé aussi modèle moyenne-variance, est le problème de programmation quadratique suivant :

$$\left\{ \begin{array}{l} \min x^T \sigma x \\ s.t. \quad r^T x \geq r^* \\ e^T x = 1 \\ x_i \geq 0, \quad \forall i \in \{1, 2, \dots, n\} \end{array} \right. \quad (1.6)$$

où :

x : est un n -vecteur, chaque élément x_i de x représente la part de capital affectée au i -ème actif,

n : est le nombre d'actifs proposés à l'investisseur,

σ : est la matrice de variance-covariance dont les composantes σ_{ij} représentent la covariance entre l'actif i et l'actif j ,

r : est un n -vecteur composé des n rendements attendus

r^* : est le rendement espéré du portefeuille,

e : est un n -vecteur de 1.

1.3.3.4 Modèle d'optimisation de portefeuille multi-objectif

L'optimisation multi-objectifs consiste généralement à concilier des objectifs conflictuels et la quête de solutions de compromis entre les objectifs tout en respectant des contraintes différentes. En conséquence, cet éventail de solutions est désigné sous le nom de Solutions optimales de Pareto.

En présence d'une version multi-critère du problème d'optimisation du portefeuille, le modèle multi-objectif (MTO) peut être modélisé sous la forme d'un problème de programmation bi-objectif.

Le but est de déterminer un assortiment efficace de portefeuilles qui maximisent et minimisent le risque simultanément. Ci-après, nous présentons quatre contraintes du monde réel : cardinalité, quantité, lot arrondi, et contrainte de pré-assignation.

$$\left\{ \begin{array}{l} \min f_1(x) = \sum_{i=1}^n \sum_{j=1}^n x_i x_j \sigma_{ij} \\ \max f_2(x) = \sum_{i=1}^n x_i \mu_i \\ s.t. \quad \sum_{i=1}^n x_i = 1 \quad (C_1) \\ \sum_{i=1}^n y_i = K \quad (C_2) \\ x_i = z_i v_i, \quad z_i \in \mathcal{Z}^+, \quad i = 1, \dots, n \quad (C_3) \\ \alpha_i y_i \leq x_i \leq \beta_i y_i \quad i = 1, \dots, n \quad (C_4) \\ y_i \geq z_i \quad i = 1, \dots, n \quad (C_5) \\ y_i, z_i \in \{0, 1\}, \quad i = 1, \dots, n \quad (C_6) \end{array} \right. \quad (1.7)$$

où :

x_i : est la variable de décision qui représente la proportion, n : est le nombre d'actifs disponibles,

μ_i : est le rendement attendu de l'actif i ,

σ_{ij} : est la covariance entre les actifs i et j .

L'équation (C_1) définit la contrainte budgétaire pour un portefeuille réalisable, autrement dit, tout le budget disponible est investi.

L'équation (C_2) définit la contrainte de cardinalité où K est le nombre d'actifs investis dans le portefeuille et y_i indique si l'actif i est investi ou non. Si y_i est égal à un, l'actif i est choisi pour être investi. Et la part du capital x_i en $[\alpha_i, \beta_i]$, où $0 \leq \alpha_i \leq \beta_i \leq 1$ sinon, l'actif i n'est pas investi et x_i est égal à zéro.

L'équation (C_3) définit la contrainte du lot rond lorsque z_i une variable entière positive et v_i est le lot minimum qui peut être sélectionné pour chaque actif. L'inclusion de la contrainte du lot rond peut rendre impossible la satisfaction exacte de la contrainte budgétaire.

Les inéquations (C_4) et (C_5) définissent la contrainte de pré assignation pour remplir les exigences subjectives des investisseurs lorsque le vecteur binaire z_i , qui indique si l'actif i est dans l'ensemble pré-assigné, qui doit être inclus dans le portefeuille ou non.

1.4 Méthode de résolution

Les méthodes de résolution exactes ou approchées proposées au cours de ces dernières années sont nombreuses, et sont le reflet de l'éventail des méthodes dont on dispose pour traiter les problèmes d'optimisation combinatoire. Nous nous proposons d'en étudier les principales.

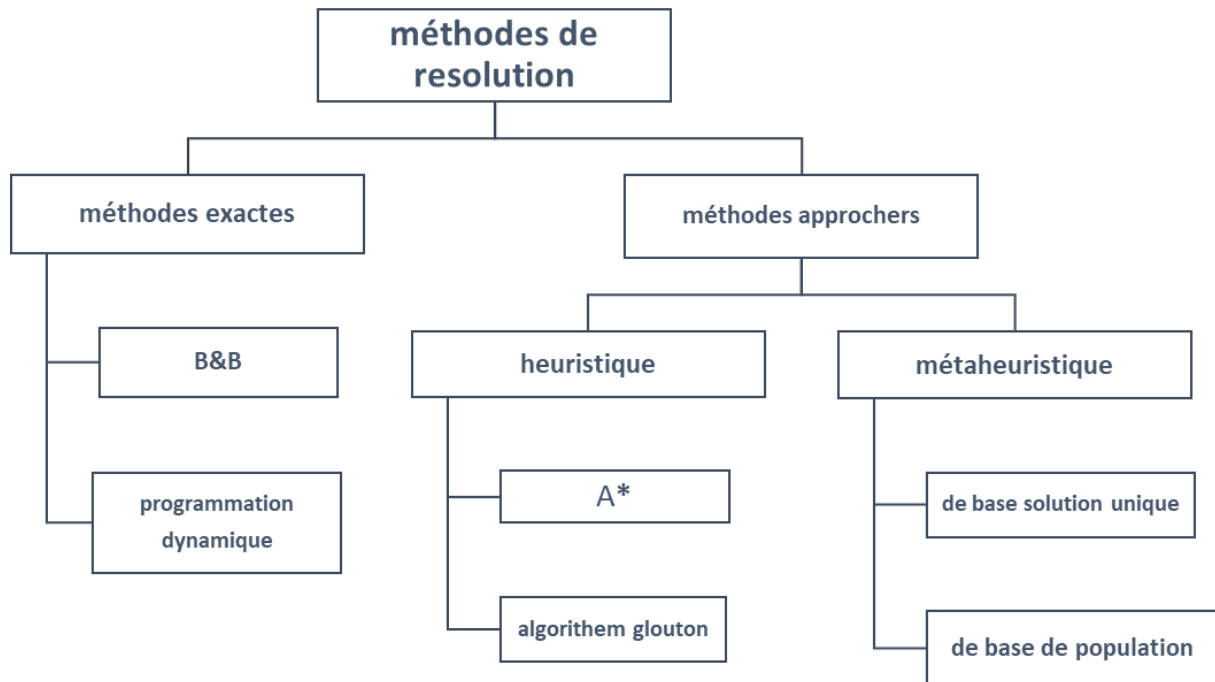


FIGURE 1.3 – Taxonomie des méthodes de résolution

1.4.1 Les méthodes exactes (complètes)

1.4.1.1 La programmation dynamique

La programmation dynamique occupe une très importance dans méthodes exactes pour la résolution une catégorie particulière de problèmes d'optimisation polynomiaux (et non N P-difficiles) sous contrainte, a été inventé en 1950 Richard Bellman.[9] La programmation dynamique est une méthode ascendante : En règle générale, nous commençons par les sous problèmes les plus petits et ensuite, nous passons aux sous problème de plus en plus difficiles.[10]

La conception d'un algorithme de programmation dynamique se décompose en quatre étapes :

1. Caractérisation de la structure d'une solution optimale.
2. Définition récursive de la valeur de la solution optimale.
3. Calcul ascendant de la valeur de la solution optimale.
4. Construction de la solution optimale à partir des informations obtenues à l'étape précédente.

Généralement faciles à implémenter les algorithmes basés sur la programmation dyna-

mique et sont très efficaces pour résoudre les problèmes de petites et moyennes tailles. Cependant, la programmation dynamique ne s'applique pas pour tous les types de problèmes d'optimisation combinatoire et son application sur des instances de grandes tailles est généralement très coûteuse voire impossible pour des raisons de complexité spatiale et/ou temporelle

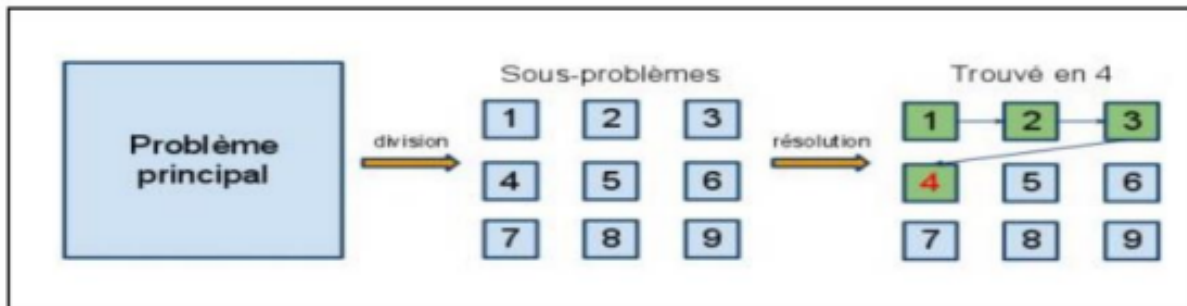


FIGURE 1.4 – Divisé en sous-problèmes

1.4.1.2 La méthode Branch and bound (B&B)

La méthode par séparation et évaluation (nommée Branch and Bound en anglais), cette méthode est très intelligente, grâce à un mécanisme qui permet de bien exploiter l'espace de recherche et l'aboutissement à la solution optimale plus rapidement que d'autres méthodes exactes en combinant deux principes primordiaux : la séparation et l'évaluation [11] **B&B** c'est une méthode constructive qui utilise l'énumération implicite basée sur la notion de bornes (supérieure et une autre inférieure) pour éviter l'énumération de larges classes de mauvaises solutions, qui est basé sur la technique de diviser pour régner, deux concepts fondamentaux : **le branchement** (séparation) qui consiste à dissocier l'espace des solutions en sous problèmes de manière à représenter le problème sous forme d'une arborescence pour les optimiser chacun individuellement (chaque nœud correspond à une solution partielle); et **l'évaluation** qui consiste à déterminer l'optimum de l'ensemble des solutions réalisables associé au nœud x_i où nous montrons que sa qualité est supérieure à la borne supérieure, le sous arbre en question sera élagué; sinon, le nœud sera divisé en sous nœud, tant qu'il reste des branches non parcourues ce processus continu jusqu'à trouver la solution optimale si elle existe [12]

L'utilisation de la méthode **B&B** nécessite :

- Une solution initiale permettant d'entamer la recherche.
- Une stratégie permettant la division du problème P en sous problèmes P_i .
- Une fonction permettant le calcul des différentes bornes.
- Une stratégie de parcours de l'arbre : parcourir en profondeur, en largeur...etc.

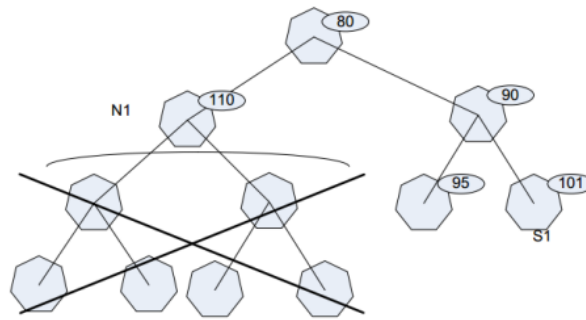


FIGURE 1.5 – Méthode de séparation/évaluation.

1.4.2 Les méthodes approchées (incomplètes)

1.4.2.1 Les méthodes heuristique

Les méthodes heuristiques sont des méthodes spécifiques pour but de fournir des solutions réalisables pour des problèmes NP-difficiles. Ce sont des règles empiriques, désireuses d'améliorer les recherches futures à travers l'expérience et les résultats qui ont été obtenus auparavant. Fait intéressant, que leurs durées de calcul sont souvent très inférieures à celles des méthodes exactes, ce qui leur permet de traiter des instances de grande taille, puisqu'elles ne sont souvent que de complexité polynomiale[13]

Algorithmes gloutons

L'idée d'un algorithme glouton (greedy algorithm en anglais) est d'approximer la solution à un problème donné, ceci est basé sur une succession de solutions localement optimales, cela signifie que à partir d'une solution initiale vide et d'ajouter d'une manière incrément-ale des composants de solutions sans remettre en cause les choix antérieurs jusqu'à obtenir une solution complète. Les composants de solutions sont ajoutés d'une manière aléatoire dans le cas le simple, mais pour l'obtention de meilleurs résultats en utilisant le critère gradient (une heuristique du bénéfice ce qu'apporte le composant à ajouter), malgré que il donne des bons résultats, il existe un inconvénient dans les premières étapes peut mener à des déplacements très faibles dans les dernières phases de construction de solutions et engendrer des solutions de qualité médiocre.[14]

Chaque fois que les algorithmes gloutons à exploiter les connaissances du problème que la performance augmentée, cela dépend étroitement de la pertinence de l'heuristique utilisée.

Pour illustrer cela, considérons le problème du voyageur de commerce, que nous avons vu dans ce chapitre.[15]

Plutôt que de chercher une solution exacte, nous pouvons regarder la solution suivante : toujours se diriger vers le nœud le plus proche, c'est-à-dire choisir l'arête de poids minimal. Il s'agit d'un algorithme glouton : en chaque point, on minimise la distance à parcourir en un pas de temps. On a donc une décision localement optimale .[16]

Toutefois, pour bien se rendre compte que cette solution n'est pas exacte, considérons le graphe suivant, pour lequel on déroule tous les chemins possibles :

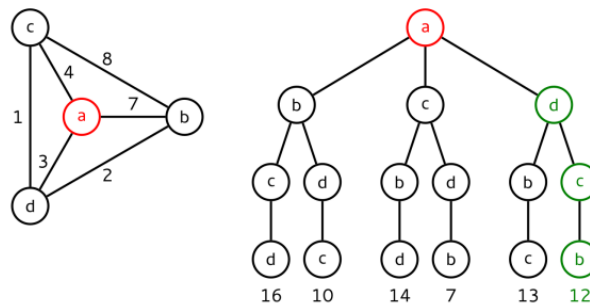


FIGURE 1.6 – Exemple d’algorithme glouton pour répondre au problème du voyageur de commerce.

Algorithme A^*

L’algorithme de recherche A^* est un algorithme de recherche de chemin dans un graphe entre un noeud initial et un noeud final tous deux donnés. Son but principal est que la première solution trouvée soit l’une des meilleures, c’est pourquoi il est célèbre dans des applications comme les jeux vidéo privilégiant la vitesse de calcul sur l’exactitude des résultats. Cet algorithme a été proposé pour la première fois par Peter E. Hart, Nils John Nilsson et Bertram Raphaelen 1968 Hart et al.[17] Il s’agit d’une extension de l’algorithme de Dijkstra de 1959.

Le principe de cet algorithme est ce qu’on appelle la recherche du premier meilleur noeud ("Best First Search"). Intuitivement, il s’agit pour un graphe donnée et à partir d’un noeud de départ, de choisir toujours le noeud qui minimise un coût (le coût est généralement connu préalablement, par exemple une distance euclidienne), jusqu’à ce qu’on tombe sur le noeud d’arrivée Zeng et Church. Le problème donc de cette approche et qu’on peut tomber sur un noeud qui ne mène vers rien, et on fera des retours arrières qui fausseront le plus court chemin par la suite.[18]

1.5 Conclusion

Dans ce chapitre, nous avons essayé de rassembler un état de l’art de quelques notions de base en optimisation combinatoire. Nous avons présenté le rôle de la théorie de la complexité dans la classification des différents problèmes selon leurs complexités en problèmes P, problème NP, problèmes NP- difficile et problèmes NP- complet, jusqu’à nous avons expliqué les notions problème combinatoire et problème d’optimisation de portefeuille , Nous avons essayé de présenter quelques éléments théorique et concept de base de portefeuille En suit nous avons présentons la théorie moderne de portefeuille a savoir le modèle de Markowitz et ses variant Enfin, nous avons mentionné les méthodes de résolution de ces problèmes, en donnant une importance particulière aux méthodes exactes et les heuristiques dans les méthodes approchées.

2.1 Introduction

Les méta-heuristiques construisent une solution moins exigeante. En fait, elles sont applicables sur une grande variété de problèmes d'optimisation de différentes complexités. En outre, elles permettent de fournir des solutions de très bonne qualité (pas nécessairement optimales) en temps de calcul raisonnable. La majorité des méta-heuristiques sont inspirées des systèmes naturels, nous pouvons citer à titre d'exemple : le recuit simulé qui est inspiré d'un processus métallurgique, les algorithmes évolutionnaires et les algorithmes génétiques qui sont inspirés des principes de l'évolution Darwinienne et de la biologie, la recherche tabou qui s'inspire de la mémoire des êtres humains, les algorithmes basés sur l'intelligence d'essaim comme l'algorithme d'optimisation par essaim de particules et l'algorithme FireFly

2.2 Les méta-heuristiques à base de solution unique

Les méta-heuristiques à base de solution unique débutent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante. En fait, la solution initiale subit une série de modifications en fonction de son voisinage. Le but de ces modifications locales est d'explorer le voisinage de la solution actuelle afin d'améliorer progressivement sa qualité au cours des différentes itérations. Le voisinage de la solution s'englobe l'ensemble des modifications qui peuvent être effectuées sur la solution elle-même. La qualité de la solution finale dépend particulièrement des modifications effectuées par les opérateurs de voisinages. En effet, les mauvaises transformations de la solution initiale mènent la recherche vers la vallée de l'optimum local d'un voisinage donné (peut être un mauvais voisinage) ce qui bloque la recherche en fournissant une solution de qualité insuffisante. De nombreuses méthodes à base de solution unique ont été proposées dans la littérature. Parmi lesquelles : la descente, le recuit simulé, la recherche tabou, la recherche à voisinage variable (VNS : Variable Neighbourhood Search), la recherche locale réitérée (ILS : Iterated Local Search), la recherche locale guidée (GLS : Guided Local Search)...etc

2.2.1 La recherche locale simple (la descente)

La recherche locale simple ou la descente est un algorithme d'amélioration très ancien. Son principe consiste à explorer le voisinage de la solution courante afin d'améliorer sa qualité progressivement, comme le montre la figure 2.1 qui représente un schéma d'évolution d'une recherche locale simple. A chaque itération du processus d'amélioration, l'algorithme modifie un ensemble de composantes de la solution courante pour permettre le déplacement vers une solution voisine de meilleure qualité. Le processus est répété itérativement jusqu'à la satisfaction du critère d'arrêt. Il est à noter qu'il existe trois types de la descente : la descente déterministe, la descente stochastique et la descente vers le premier meilleur voisin. L'algorithme 1 résume les étapes de l'algorithme général de la descente. L'algorithme entame la recherche par la construction d'une solution initiale s et l'évaluation de sa qualité $f(s)$. Ensuite il commence l'ensemble des étapes du processus d'amélioration suivantes :

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

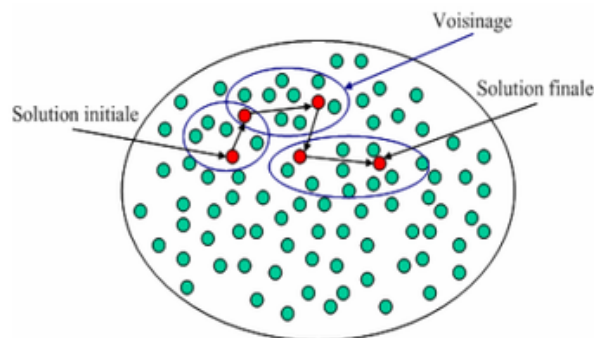


FIGURE 2.1 – Un schéma d'évolution d'une recherche locale simple [19]

Le processus d'amélioration sera répété jusqu'à arriver au cas où toutes les voisines candidates sont de piètre qualité que la solution courante. Autrement dit, la recherche s'arrête lorsqu'un optimum local est atteint. L'avantage de la recherche locale simple revient à sa simplicité et à sa rapidité. Cependant, son problème réside dans le fait d'être bloquée par le premier optimum local rencontré. Ce dernier n'est pas forcément l'optimum global. Il peut être très loin de l'optimum global.

2.2.2 Le recuit simulé

L'algorithme du recuit simulé a été proposé par Kirkpatrick, Gelatt et Vecchi [20]. Son principe se base sur la procédure du recuit des métaux utilisée par les métallurgistes. Ces derniers chauffe à blanc le métal, puis ils laissent l'alliage se refroidir très lentement afin d'aboutir à un alliage sans défauts. En fait, les thermodynamiciens ont remarqué qu'une baisse brutale de la température d'un liquide entraîne une reproduction d'un optimum

Algorithme 1 : La recherche locale simple (la descente)**Début**

Construire une solution initiale s ;

Calculer la fitness $f(s)$ de s ;

tant que *la condition d'arrêt n'est pas vérifiée* **faire**

 Modifier s pour obtenir une nouvelle solution voisine s' ;

 Calculer $f(s')$;

Si $f(s')$ est meilleure que $f(s)$ **alors**

 Remplacer s par s' ;

Fin Si

fin

Retourner s ;

Fin

local, i.e. une structure amorphe. Alors qu'une baisse progressive de la température du liquide permet d'aboutir à un optimum global, i.e. une structure bien construite. C'est l'idée prise en considération par les métallurgistes qui savent que si le métal refroidit trop vite, il contiendra beaucoup de défauts microscopiques et s'il refroidit lentement ils obtiendront une structure bien ordonnée. La méta-heuristique du recuit simulé s'inspire de l'algorithme de Métropolis [21], dont le principe (pour un problème de maximisation) peut être résumé comme suit :

- Entamer la recherche avec une solution initiale s ;
- Affecter une valeur initiale à la température T ;
- Calculer la fitness $f(s)$ de la solution initiale s .
- Générer une solution s' voisine de s ;
- Calculer la fitness $f(s')$ de s' ;
- Calculer l'écart de qualité (fitness) entre la solution s et la solution s' comme suit

$$\Delta(f) = f(s') - f(s)$$

- **Si** $\Delta(f) \geq 0$ **alors** $s \leftarrow s'$
- **Sinon** générer un nombre aléatoire $r \in [0, 1]$;
- **Si** $r < \exp^{\Delta(f)/T}$ **alors** $s \leftarrow s'$.

Un schéma général de l'algorithme du recuit simulé (RS) est présenté dans l'algorithme(). Le recuit simulé permet d'accepter une solution de piètre qualité que la solution courante afin de diversifier la recherche et échapper au piège de l'optimum local. Le fait d'accepter des solutions de mauvaises qualités peut mener la recherche vers la meilleure solution (l'optimum global) car cette dernière peut faire partie du voisinage d'une mauvaise solution et non pas d'une bonne solution (i.e. la solution courante qui est de meilleure qualité que sa voisine) qui peut représenter un optimum local. Cependant, l'acceptation de solutions de mauvaises qualités peut causer une perte de la meilleure solution rencontrée au cours de la recherche et entraîne une convergence vers une solution de mauvaise qualité qu'une autre déjà trouvée. Ce problème peut être facilement résolu en ajoutant une variable permettant la mémorisation de la meilleure solution trouvée

Algorithme 2 : Le recuit simulé**Début**Construire une solution initiale s ;Calculer la fitness $f(s)$ de s ;Initialiser une valeur de la température T ;**tant que** *la condition d'arrêt n'est pas vérifiée* **faire** Générer une solution s' voisine s ; Calculer $f(s')$; Calculer $\Delta(f) = f(s') - f(s)$; **Si** $\delta(f) \geq 0$ **alors** // cas de maximisation $s_{best} = s'$; $s = s'$ **Sinon Si** $r < \exp\left(\frac{\Delta(f)}{T}\right)$ **alors** $s = s'$; **Fin Si** Décroître la température T ;**fin**Retourner s_{best} ;**Fin**

L'acceptation d'une solution de mauvaise qualité est établi en fonction de deux facteurs : l'écart de qualité entre la solution courante et sa voisine d'un coté, et la température de l'autre coté. Plus la température est élevée, plus la probabilité d'accepter des solutions de mauvaises qualités est forte. La valeur initiale de la température T décroît au cours de la recherche pour tendre vers le 0. Ce paramètre (i.e. la température) a un effet non négligeable sur la performance de l'algorithme. Il doit être soigneusement ajusté tout au long de la recherche. En fait, un refroidissement rapide de la valeur de T peut entraîner une convergence prématurée de l'algorithme vers un optimum local de mauvaise qualité. Tandis qu'un refroidissement lent peut mener la recherche vers une solution de bonne qualité. Cependant, le refroidissement trop lent nécessite un temps de calcul élevé. En outre, le choix de la valeur initiale de T joue un rôle primordial dans le processus de recherche. Il dépend de la qualité de la solution initiale. Si cette dernière est choisie aléatoirement, il sera préférable d'affecter une valeur élevée à T pour donner plus de chance d'aboutissement à de bonnes solutions. Si le choix de la valeur initiale de la solution est expérimental ou empirique, la température initiale peut être basse.

Le recuit simulé est un algorithme basé sur la recherche à voisinage (recherche locale). c'est un algorithme simple, facile à implémenter et à adapter à un grand nombre de problèmes : traitement d'image, sac à dos, voyageur de commerce, ordonnancement, etc. Comparé avec la recherche locale simple, le RS permet de se sauver du piège de l'optimum local et d'offrir des solutions de bonne qualité comme il présente une souplesse d'intégration des contraintes liées au problème traité. En revanche, cet algorithme dispose d'un nombre important de paramètres (température initiale, paramètres liés à la fonction d'ajustement de la température... etc.) à ajuster. En outre, le RS est un algorithme lent surtout avec les problèmes de grande taille

2.2.3 La recherche à voisinage variable

La recherche à voisinage variable est une méthode basée sur la recherche locale. Elle a été introduite en 1997 par Mladenovic et Hansen [24]. La recherche à voisinage variable est notée par RVV ou encore VNS (de l'anglais : Variable Neighbourhood Search). Elle se base sur un simple principe de recherche fondé sur le changement systématique du voisinage. A l'opposé d'autres méthodes basées sur la recherche locale, la VNS n'utilise pas un seul voisinage pour exploiter la solution courante. Par contre, elle utilise plusieurs voisinages dans un ordre prédéfini. Son principe de base consiste à entamer la recherche avec un voisinage V_1 d'une solution initiale s . Lorsque la recherche se bloque dans un optimum local, on continue la recherche avec un autre voisinage puis on revient au premier voisinage après chaque amélioration. La recherche continue en passant d'un voisinage à un autre à chaque blocage dans la vallée de l'optimum local du voisinage utilisé.

L'idée de parcourir un ensemble de voisinage permet de sauver la recherche du blocage causé par le piège de l'optimum local. Cette stratégie permet de diversifier l'exploration de l'espace de recherche et donne au processus de recherche plus de chances de rencontrer l'optimum global. Cependant, le choix et l'ordre de parcours des différents voisinages jouent un rôle non négligeable dans la performance de l'algorithme. En fait, s'il est bien étudié, la recherche converge vers la solution optimale (l'optimum global), sinon la convergence de la solution optimale ne sera pas garantie.

L'algorithme 4 représente un pseudo code résumant les étapes du schéma général de la recherche à voisinage variable. L'algorithme commence la recherche par une solution initiale s . Puis, il génère une solution s' du premier voisinage de la solution s . Après évaluation de la qualité de s et celle de s' , si s' est meilleure que s , l'algorithme remplace s par s' et il continue l'amélioration de la solution courante en générant d'autres solutions du même voisinage. Sinon, l'algorithme continue la recherche avec le deuxième voisinage jusqu'à arriver à la satisfaction du critère d'arrêt

2.3 Les méta-heuristiques à base de population de solutions

Les méta(heuristiques à base de population de solutions débutent la recherche avec une panoplie de solutions). Elles s'appliquent sur un ensemble de solutions afin d'en extraire la meilleure (l'optimum global) qui représentera la solution du problème traité. L'idée d'utiliser un ensemble de solutions au lieu d'une seule solution renforce la diversité de la recherche et augmente la possibilité d'émergence de solutions de bonne qualité. Une grande variété de méthodes basées sur une population de solutions a été proposée dans la littérature, commençant par les algorithmes évolutionnaires, passant par les algorithmes génétiques et arrivant aux algorithmes à base d'intelligence par essais (l'algorithme d'optimisation par essaim de particules, l'algorithme de colonies de fourmis, l'algorithme de colonies d'abeilles, la recherche coucou, l'algorithme d'optimisation par coucou ...) qui ont connus une investigation remarquable ces deux dernières décennies

Algorithme 3 : la recherche à voisinage variable**Début**

Définir la structure du voisinage N_k , où $k \in \{1, 2, \dots, k_{max}\}$;

Construire une solution initiale s ;

Calculer la fitness $f(x)$ de s ;

tant que la condition d'arrêt n'est pas vérifiée **faire**

$k = 1$;

tant que $k \leq k_{max}$ **faire**

 Générer une solution s' voisine de s , où $N_k \in s'$;

 Calculer $f(s')$;

Si $f(s)$ est meilleure que $f(s')$ **alors** $s = s'$

$k = 1$;

Si non $k = k + 1$;

Fin Si

fin

fin

Retourner s ;

Fin

2.3.1 Les algorithmes évolutionnaires

L'apparition des algorithmes évolutionnaires revient aux années soixante. Elles sont inspirées du principe de l'évolution naturelle des espèces biologiques. Précisément, de la théorie de l'évolution exposée par Charles Darwin [25], qui se base sur le principe de la sélection naturelle. Le principe de cette dernière stipule que les individus bien adaptés à l'environnement ont plus de chances de se reproduire et de survivre que les autres individus. En fait, la combinaison des caractéristiques des individus peut former au fil des générations de nouveaux individus de plus en plus mieux adaptés à leur environnement et qui peuvent avoir plus de chances de survivre que leurs parents.

La découverte de l'ADN et de son rôle en 1944 par Tomas Avery a renforcé le principe de la théorie d'évolution proposée par Charles Darwin. En effet, chaque cellule de l'organisme de l'individu possède des chromosomes, ces derniers sont composés des molécules d'ADN qui contiennent tout le patrimoine génétique de l'individu. La reproduction qui se base sur la combinaison du patrimoine génétique des individus, consiste à échanger des gènes entre individus. L'échange du patrimoine génétique peut contribuer dans l'amélioration des caractéristiques des individus de la nouvelle génération et donc donne plus de chances de survie aux membres de la nouvelle génération.

Les algorithmes évolutionnaires s'inspire de l'évolution naturelle des êtres vivants. Ils adoptent une sorte d'évolution artificielle pour améliorer la qualité des individus de la population. En fait, ils font évoluer itérativement une population d'individus. Ces derniers représentent des solutions du problème traité. Les qualités des individus sont mesurées à chaque itération du processus de d'évolution. En fonction de leurs qualités, les meilleurs individus seront sélectionnés pour subir des combinaisons qui permettent la production d'une nouvelle population (dite : population d'enfants). Les individus (ou une partie des individus) de la nouvelle population vont remplacer les individus de la population courante (dite : population de parents) pour construire la nouvelle génération d'individus. Le

processus d'évolution d'un algorithme évolutionnaire est basé sur trois opérations principales : la sélection, la reproduction et l'évaluation :

- **La Sélection** : Cette opération intervient dans deux phases. Elle est appliquée au premier lieu pour choisir les meilleurs individus parents qui vont se reproduire pour construire de nouveaux individus enfants. Ensuite, elle est appliquée à la fin de chaque itération pour opter pour les individus qui vont survivre et construire la nouvelle population
- **La reproduction** : Cette opération est en général composée de deux autres opérations : le croisement et la mutation. Elle permet la génération de nouveaux individus en combinant (phase de croisement) les caractéristiques des individus sélectionnés puis en appliquant quelques modifications de certains individus (phase de mutation) pour améliorer leurs qualités
- **L'évaluation** : Cette opération consiste à mesurer la qualité de chaque individu (calculer la fitness des individus). La figure 2.3 représente un diagramme résumant les étapes générales d'un algorithme évolutionnaire.

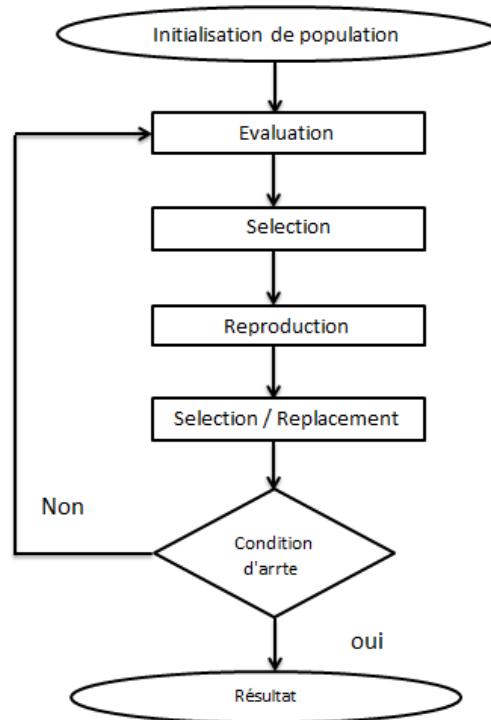


FIGURE 2.2 – Organigramme d'un algorithme évolutionnaire

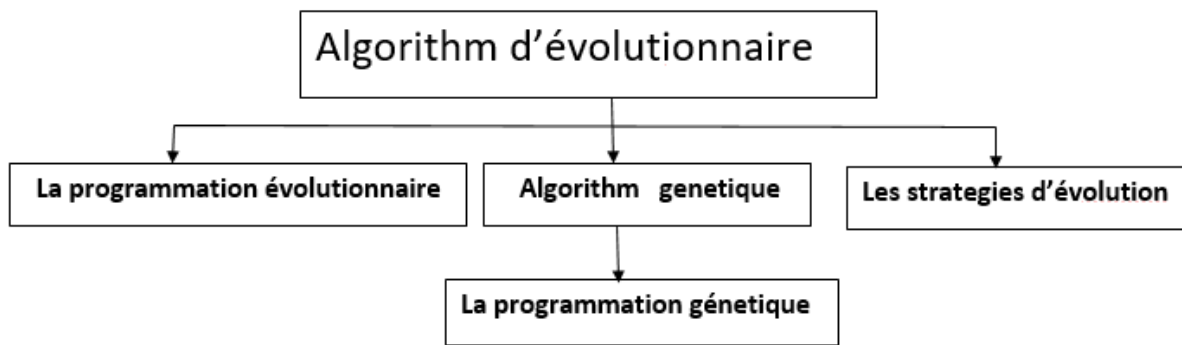


FIGURE 2.3 – Les types des algorithmes évolutionnaires.

Les algorithmes évolutionnaires forment une classe principale de trois sous classes d'algorithmes (voir figure 2.3) : les stratégies d'évolution, la programmation évolutionnaire et les algorithmes génétiques.

Les stratégies d'évolution : [26] Ont été conçues pour la résolution des problèmes d'optimisation continus.

La programmation évolutionnaire : [27] Les algorithmes de la classe de la programmation évolutionnaire sont conçus pour faire évoluer des structures d'automates à état fini.

L'algorithme génétique : [28] Il sera bien détaillé dans la section suivante. Il est à noter qu'il existe une sous classe de la classe des algorithmes génétiques appelée « La programmation génétique » [29]. Cette dernière utilise des structures arborescentes pour représenter les individus de la population.

2.3.2 L'algorithme génétique

L'algorithme génétique représente une célèbre méta-heuristique évolutionnaire. Il a été proposé par Jhon Holland en 1975 [30]. L'algorithme génétique s'inspire des mécanismes biologiques tels que les lois de Mendel et la théorie de l'évolution proposée par Charles Darwin [31]. Son processus de recherche de solutions à un problème donné imite celui des êtres vivants dans leur évolution. Il utilise le même vocabulaire que celui de la biologie et la génétique classique, on parle donc de : gène, chromosome, individu, population et génération.

Un gène : est un ensemble de symboles représentant la valeur d'une variable. Dans la plupart des cas, un gène est représenté par un seul symbole (un bit, un entier, un réel ou un caractère).

Un chromosome : est un ensemble de gènes, présentés dans un ordre donné de manière qui prend en considération les contraintes du problème à traiter. Par exemple, dans le problème du voyageur de commerce, la taille du chromosome est égale au nombre de villes à parcourir. Son contenu représente l'ordre de parcours de différentes villes. En outre, on

doit veiller à ce qu'une ville (représentée par un nombre ou un caractère par exemple) ne doit pas figurer dans le chromosome plus qu'une seule fois.

Un individu : est composé d'un ou de plusieurs chromosomes. Il représente une solution possible au problème traité.

Une population : est représentée par un ensemble d'individus (i.e. l'ensemble des solutions du problème).

Une génération : est une succession d'itérations composées d'un ensemble d'opérations permettant le passage d'une population à une autre.

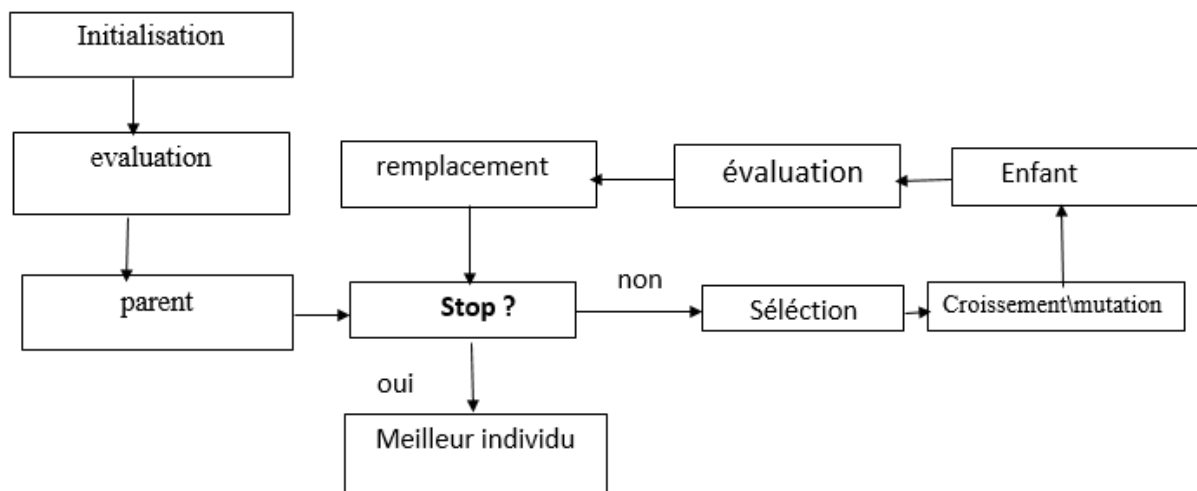


FIGURE 2.4 – Démarche d'un algorithme génétique. [32]

L'algorithme génétique fait évoluer une population composée d'un ensemble d'individus pendant un ensemble de génération jusqu'à ce qu'un critère d'arrêt soit vérifié. Le passage d'une population à une autre est réalisé grâce à des opérations d'évaluation, de sélection, de reproduction (croisement et mutation) et de remplacement.

L'algorithme commence la recherche avec un ensemble d'individus. A chaque itération de la procédure de recherche, les meilleurs individus sont sélectionnés pour survivre et se reproduire. La sélection des individus est fondée sur leurs qualités qui sont mesurées à partir d'une fonction appelée « fonction objectif ou fonction fitness ». Ensuite, les individus (appelés parents) sont sélectionnés pour subir des opérateurs de croisement et de mutation permettant la génération d'une autre population d'individus (appelés enfants). Les individus de la nouvelle population seront évalués pour remplacer une partie des individus de la population courante. La figure 2.4 illustre un schéma général des étapes du processus de recherche de l'algorithme génétique.

Le processus de recherche de l'algorithme génétique est fondé sur les opérateurs suivants :

Un opérateur de codage des individus : Il permet la représentation des chromosomes représentant les individus.

Un opérateur d'initialisation de la population : Il permet la production des individus de la population initiale. Malgré que cet opérateur ne intervient qu'une seule fois et au début de la recherche, mais il joue un rôle non négligeable dans la convergence vers l'optimum global. En fait, le choix de la population initiale peut rendre la recherche de la solution optimale du problème traité plus facile et plus rapide.

Un opérateur de sélection : Il permet de favoriser la reproduction des individus qui ont les meilleures fitness (i.e. les meilleures qualités).

Un opérateur de croisement : Il permet l'échange des gènes entre parents (deux parent en général), pour créer 1 ou deux enfants en essayant de combiner les bonnes caractéristiques des parents. Le but de cet opérateur est la création de nouveaux individus en exploitant l'espace de recherche.

Un opérateur de mutation : Il consiste à modifier quelques gènes des chromosomes des individus, dans le but d'intégrer plus de diversité au sein du processus de la recherche.

Un opérateur d'évaluation : Il permet de valoriser la qualité des individus, en se basant sur la fonction « objectif » (la fonction fitness) qui permet de calculer la qualité de chaque individu.

En outre des différents opérateurs permettant de guider la recherche par l'algorithme génétique, ce dernier nécessite un certain nombre de paramètres de base, sur lesquels dépendent les différents opérateurs cités en dessus. Ces paramètres doivent être fixés à l'avance, ils jouent un rôle très important dans la performance de l'algorithme. On parle de : la taille de la population, la probabilité de croisement, la probabilité de mutation et le nombre maximum de génération.

La taille de la population : Elle représente le nombre d'individus de la population. S'il est trop grand, le processus de recherche demande un coût de recherche élevé, que se soit en termes d'espace mémoire ou du temps de calcul nécessaires. Cependant, s'il est trop petit, l'algorithme risque d'être tomber dans le cas de la convergence prématurée à cause du manque de la diversité au sein de la population. Il est préférable donc de choisir une taille moyenne en prenant en considération l'instance du problème à traiter.

La probabilité de croisement : Elle représente la probabilité d'échange de patrimoine (i.e. les gènes) entre deux individus (ou plus). Plus elle est grande, plus elle permet la génération de nouveaux enfants qui peuvent être meilleurs que leurs parents.

La probabilité de mutation : Elle est en général faible, dans le but d'échapper aux possibilités de modifications radicales des solutions, particulièrement, des solutions de bonnes qualités qui ne nécessitent que peu d'amélioration pour passer aux solutions optimales.

Le nombre maximum de génération : Ce paramètre peut jouer le rôle d'un critère d'ar-

rêt. Il peut construire un obstacle pour l'algorithme. En fait, il peut empêcher les différents opérateurs d'aboutir à la meilleure solution s'il est trop petit. Comme il peut engendrer un temps de calcul prohibitif dans le cas où il est trop grand. Ainsi, le choix de sa valeur peut se baser sur des tests préliminaires. L'algorithme 5 représente l'ensemble des étapes de l'algorithme génétique.

Algorithme 4 : L'algorithme génétique

Début

Initialiser les paramètres nécessaires ;
 Initialiser une population de N individus ;
 Evaluer les N individus ;

tant que *la condition d'arrêt n'est pas satisfaite* **faire**

 Utiliser l'opérateur de sélection pour sélectionner K individus ;
 Applique l'opérateur de croisement sur K individus avec probabilité P_c ;
 Applique l'opérateur de mutation sur les individus avec probabilité P_m ;
 Utiliser l'opération d'évaluation pour évaluer les enfants obtenus ;
 Utiliser l'opération de sélection pour remplacer quelque individus parents par
 des individus enfants ;

fin

Retourner la ou les meilleures solution ;

Fin

L'algorithme génétique a été utilisé pour la résolution de nombreux problèmes académiques et/ou industriels. Son avantage principal est qu'il permet une bonne combinaison entre l'exploitation de solutions et l'exploration de l'espace de recherche. Cela est établi en fonction des opérateurs de croisement et de mutation respectivement. Cependant, son inconvénient réside dans deux points : un temps de calcul assez important pour pouvoir converger vers la solution optimale (quoique l'apparition des ordinateurs de plus en plus puissants a permis de remédier à ce problème). Et le nombre de paramètres importants (taille de la population, paramètres de sélection, paramètres de croisement, paramètres de mutation, critère d'arrêt...).

2.3.2.1 Le codage des individus

Avant de générer l'ensemble des individus représentant la population initiale, il faut d'abord penser à définir le codage convenable des individus de la population. Sachant que ces derniers sont représentés par un ou plusieurs chromosomes. Chaque chromosome est composé d'un ensemble de gènes et chaque gène exprime un paramètre ou une information. Tout ça nécessite la définition d'un codage permettant la modélisation et la manipulation des solutions (les individus) de la population. Le choix du codage joue un rôle très important. En fait, il doit permettre la représentation des différentes solutions possibles, comme il doit prendre en considération le type du problème à traiter (continu, discret, binaire ...). On distingue les types de codage suivants :

1. Le codage binaire :

Le codage binaire consiste à utiliser des bits (0 ou 1) pour représenter les différentes solutions. Dans le sens où, les gènes sont représentés par des bits et les chromosomes sont représentés par des chaînes de bits. Le type de ce codage s'adapte bien aux problèmes de type

binaire, comme le problème MAX SAT ou les problèmes du sac à dos. Prenons l'exemple du problème du sac à dos dans sa forme la plus simple (unidimensionnelle). Supposant, que nous devons sélectionner un sous ensemble d'objets parmi un ensemble de 10 objets. Les solutions peuvent être facilement représentées par des chaînes binaires de taille 10, où chaque bit représente l'état d'un objet : le bit 1 pour dire que l'objet est sélectionné, le bit 0 pour dire que l'objet est écarté. La Table 2.1 représente une des solutions du problème. Selon la figure (), les objets 2, 3, 5, 8, 9, et 10 sont sélectionnés. Tandis que les autres (i.e. 1, 4, 6 et 7) sont écartés.

0	1	1	0	1	0	0	1	1	1
---	---	---	---	---	---	---	---	---	---

TABLE 2.1 – Codage binaire d'un chromosome

2. Le codage réel :

Ce type de codage est beaucoup plus efficace pour représenter des problèmes de type continu. Il représente les solutions par des suites de type réel, comme le montre la figure

0.23	1.25	10.5	0.48	3.00	-20.87
------	------	------	------	------	--------

TABLE 2.2 – Codage réel d'un chromosome

On peut distinguer un autre type spécial du codage réel, c'est le codage entier ou discret. Il utilise des entiers au lieu de réels. Ce type est beaucoup plus efficace pour la représentation de certain type de problèmes discrets, comme les problèmes d'ordonnancement et le problème du voyageur de commerce. Prenons l'exemple du problème du voyageur de commerce où un voyageur de commerce doit parcourir un ensemble de 10 villes en choisissant le plus court chemin permettant de passer par chacune une seule fois. La Table 2.2 représente une solution du problème où chaque gène représente une ville et l'ensemble des gènes représente l'ordre de parcours de l'ensemble des 10 villes.

2	6	3	7	5	4	10	8	1	9
---	---	---	---	---	---	----	---	---	---

TABLE 2.3 – Codage entier d'un chromosome

3. Le codage à caractère :

Il s'agit d'utiliser une suite de caractères différents pour représenter le chromosome, comme le montre la figure

A	V	-	G	L	M
---	---	---	---	---	---

TABLE 2.4 – Codage à caractère d'un chromosome

4. Le codage arborescent :

Ce type de codage utilise une représentation arborescente des individus avec un ensemble de nœuds, où l'ensemble des nœuds représente un individu, comme le montre la figure

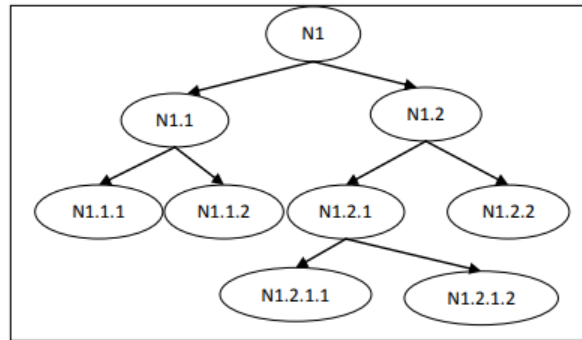


FIGURE 2.5 – Codage arborescent d'un chromosome

2.3.2.2 La sélection

La sélection joue un rôle très important dans le processus de recherche de l'algorithme génétique. En effet, elle intervient dans deux étapes de chaque itération :

Au début de chaque itération pour sélectionner les individus parents qui vont se reproduire entre eux.

A la fin de chaque itération pour sélectionner les individus enfants qui vont remplacer les individus parents dans le but de créer une nouvelle population en respectant la taille de la population pour ne pas entraîner une explosion démographique de la population.

La sélection des individus dépend essentiellement de leurs qualités (fitness). En effet, les meilleurs individus sont sélectionnés pour se reproduire, survivre et donc remplacer les moins performants. On peut distinguer plusieurs types de sélections comme : la sélection à la roulette, la sélection par tournoi et la sélection élitiste [33]

2.3.2.3 Le croisement

Le croisement consiste à générer un ou deux nouveaux individus à partir d'un couple de parents choisi par l'opérateur de sélection. C'est un opérateur qui permet la création de nouveaux individus en combinant les gènes des individus parents. Ainsi, après répartition de l'ensemble des individus (par sélection en utilisant l'opérateur de sélection) en couples, ces derniers subissent un opérateur d'échange de gènes permettant la génération de nouveaux individus. Cet échange est souvent fait au hasard, mais il permet des fois de remplacer les mauvais gènes d'un parent par les bons gènes de l'autre parent. Cela permet de créer des enfants de bonne qualité que leurs parents. On distingue deux types essentiels d'opérateurs de croisement : le croisement n-point et le croisement uniforme.

Le croisement n-point :

Ce type de croisement consiste à choisir n-points de coupures ($n = 1, 2, \dots$), puis échanger les fragments de gènes délimités par les points de coupure choisis. On distingue dans ce type deux autres types de croisement qui sont très utilisés : le croisement 1-point et le croisement 2-points.

Le croisement 1-point :

consiste à choisir un seul point de coupure, puis échanger les fragments situés après ce

point de coupure. Comme le montre la figure

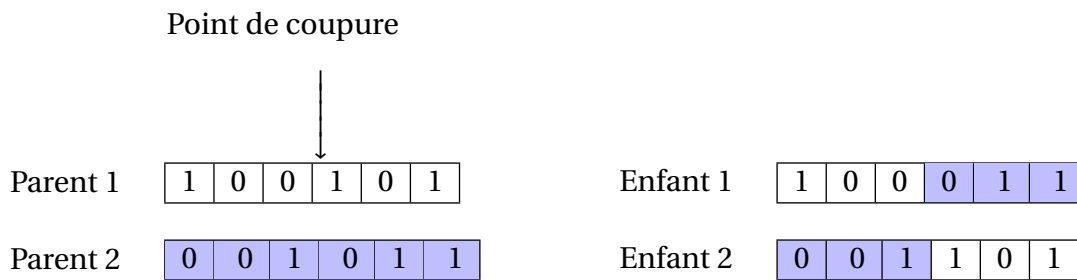


TABLE 2.5 – Croisement 1-point.

Le croisement 2-point :

consiste à choisir deux points de coupure, puis échanger les fragments situés entre ces deux points comme le montre la figure

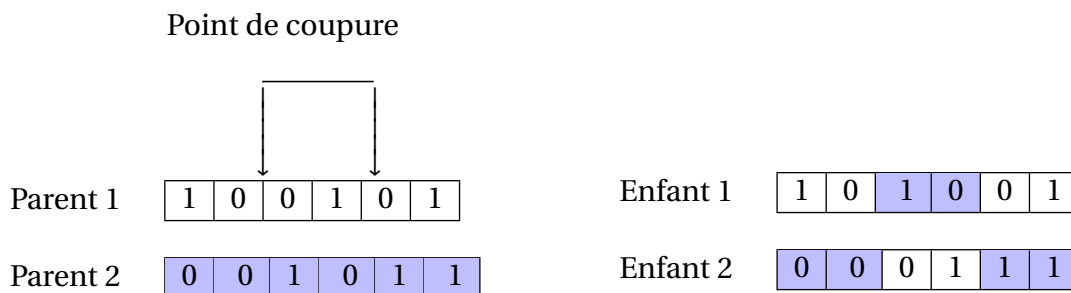


TABLE 2.6 – Croisement 2-point.

Le croisement uniforme :

Ce type de croisement est fondé sur la probabilité. En fait, il permet la génération d'un enfant en échangeant chaque gène des deux parents avec une probabilité égale à 0.5, comme le montre la figure

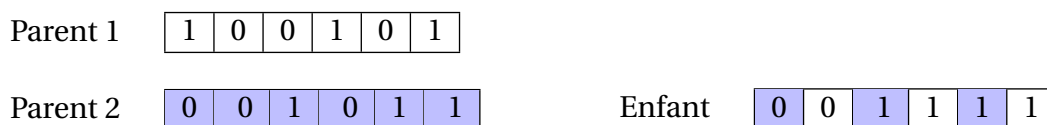


TABLE 2.7 – Croisement uniforme.

2.3.2.4 La mutation

L'opérateur de mutation permet d'apporter des modifications partielles (légères) aux chromosomes avec une certaine probabilité très faible. L'objectif de l'opérateur de mutation est d'aider l'algorithme à échapper au problème de stagnation de la recherche causé par les optima locaux. Il permet d'explorer l'espace de recherche en apportant à l'algorithme la possibilité de couvrir la totalité de l'espace de recherche [34]. L'opérateur de mutation dépend du type de codage utilisé. Les opérateurs de mutation les plus utilisés sont : la mutation Bitflip et la mutation 1-bit [35].

2.3.2.5 Le remplacement

Après croisement et mutation des individus, le nombre d'individus de la population augmente et la taille de cette dernière dépasse sa limite. En fait, la phase de reproduction (croisement et mutation) permet de créer une nouvelle population composée de deux groupes d'individus : parents et enfants. La phase de remplacement permet de décider quels sont les individus qui vont représenter la nouvelle population. Parmi les opérateurs de remplacement existant, nous distinguons le remplacement stationnaire et le remplacement élitiste [36].

2.3.2.6 La recherche par harmonies

La recherche par harmonies (HS : Harmony Search) est une très récente méta-heuristique. Elle a été proposée par Geem et ses collègues [37]. A l'opposé des autres méta-heuristiques qui s'inspirent des phénomènes naturels, la recherche par harmonies s'inspire du processus de recherche de la meilleure harmonie musicale. Les étapes du processus de recherche de l'algorithme de recherche par harmonies sont résumées dans l'algorithme 6 et expliquées dans ce qui suit

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

- la taille de la mémoire d'harmonies (i.e. la population), notée par HMS (de l'anglais : Harmony Memory Size).
- Le taux de considération de la mémoire harmonique, noté par HMCR (de l'anglais : Harmony Memory Considering Rate), dont le rôle est de décider si la mémoire HM sera utilisée ou non.
- Le paramètre PAR (de l'anglais : Pitch Adjusting Rate), représentant la probabilité d'apporter quelques modifications à un élément de la HM.
- Le critère d'arrêt (généralement un nombre maximum d'itérations).

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

Ensuite, l'algorithme passe à l'étape d'amélioration des harmonies. Cette étape consiste à améliorer une solution $x'_i = x'_1, x'_2, \dots, x'_N$ en se basant sur trois règles : la considération de la mémoire HM, l'ajustement des valeurs des variables de la solution et la sélection aléatoire. Après génération du nouveau vecteur (nouvelle solution) $x'_i = x'_1, x'_2, \dots, x'_N$; les composantes obtenues par considération de la mémoire HM sont examinées pour décider s'ils devront être ajustées ou non. Suit à l'étape d'amélioration de solutions, le HS passe à l'étape de mise à jour de la mémoire HM. Cette étape consiste à remplacer la mauvaise solution de la matrice HM par la nouvelle solution trouvée si cette dernière est de meilleure qualité (comparée avec la mauvaise solution). Les étapes d'amélioration et de mise à jour seront répétées jusqu'à la satisfaction du critère d'arrêt.

Algorithme 5 : L'algorithme général de la recherche par harmonies

Début

Initialiser les paramètres nécessaires;

Initialiser la mémoire HM (une population d'harmonies);

tant que *la condition d'arrêt n'est pas satisfaite* **faire** Produire une nouvelle solution en améliorant la solution x'_i ;

Mettre à jour la mémoire HM;

fin

Retourner la ou les meilleures solutions;

Fin

2.4 les algorithmes à base de l'intelligence par essaim

2.4.1 L'intelligence par essaim de particules

Les algorithmes basés sur l'intelligence par essaim forment une branche d'algorithme inspirés des phénomènes naturels. Ces algorithmes s'inspirent généralement des comportements collectifs de certaines espèces dans la résolution de leurs problèmes, pour développer des méta-heuristiques permettant la résolution de différents problèmes d'optimisation. Le mot « essaim » est généralement utilisé pour désigner un ensemble fini de particules ou d'agents interactifs. Les oiseaux évoluant en groupes, les bancs de poissons, les colonies de fourmis, les colonies d'abeilles et même les systèmes immunitaires sont des exemples d'essaim. Les oiseaux évoluant en groupe forment des essaims dont les particules sont des oiseaux, les bancs de poissons forment des essaims dont les particules sont des poissons, les colonies de fourmis forment des essaims dont les particules sont des fourmis, les colonies d'abeilles forment des essaims dont les particules sont des abeilles, le système immunitaire forme un essaim de particules représenté par des cellules de reconnaissance et de protection. Ainsi, en imitant le comportement social des particules formant des essaims capable de l'auto-organiser, plusieurs algorithmes ont été proposés ces dernières décennies comme : L'optimisation par essaim de particules, le système immunitaire artificiel, les colonies de fourmis artificielles, les colonies d'abeilles artificielles, la recherche coucou, l'algorithme d'optimisation par coucou...etc.

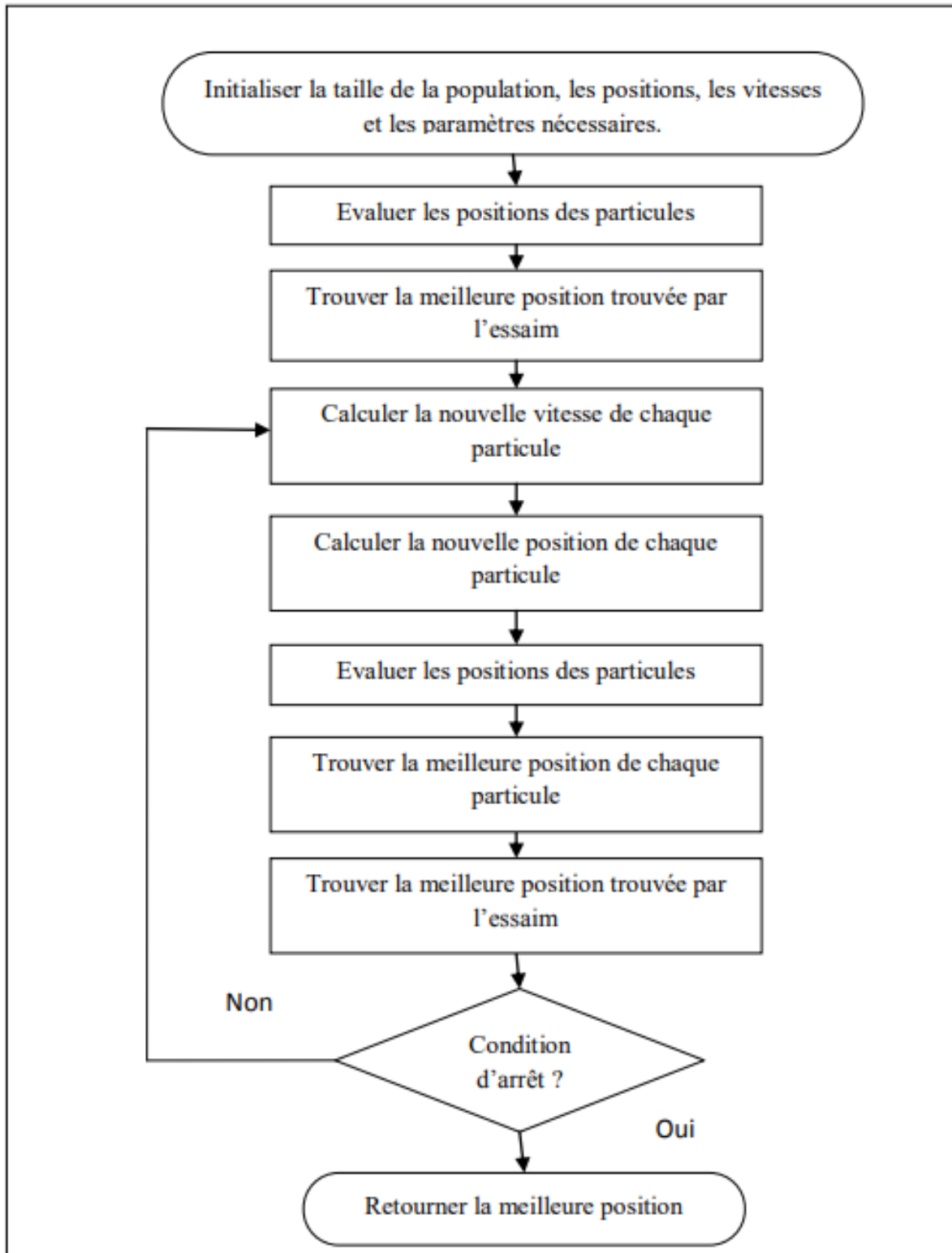


FIGURE 2.6 – Organigramme de l'algorithme d'optimisation par essaim de particules.

L'optimisation par essaim de particules : L'optimisation par essaim de particules est une méta-heuristique populaire basée sur l'intelligence par essaim. Elle a été proposée en 1995 par Kennedy et Eberhart. L'optimisation par essaim de particules s'inspire du comportement social des oiseaux évoluant en groupe et des bancs de poissons. L'algorithme d'optimisation par essaim de particule lance la recherche avec une population de solutions, où chacune est appelée « particule ». Cette dernière est caractérisée par une vitesse de déplacement et une position dans l'espace de recherche. Au cours du processus de la recherche, chaque particule se déplace pour modifier sa position dans l'espace de recherche en fonction de sa vitesse actuelle, sa position actuelle, sa meilleure position trouvée au cours des itérations passées et la meilleure position trouvée par l'essaim. Son déplacement lui permet de mettre à jour sa position et sa vitesse de déplacement à chaque itération. Les étapes principales de l'algorithme d'optimisation par essaim de particules sont présentées dans l'organigramme présenté dans la figure 2.6 Plus de détail sur l'algorithme d'optimisation par essaim de particules est présenté dans le chapitre 3.

2.4.2 Algorithme FireFly

2.4.2.1 Inspiration

Les lucioles (en anglais FireFly) sont de petits coléoptères ailés capables de produire une lumière clignotante froide pour une attraction mutuelle. Les femelles peuvent imiter les signaux lumineux d'autres espèces afin d'attirer des mâles qu'elles capturent et dévorent. Les lucioles ont un mécanisme de type condensateur, qui se décharge lentement jusqu'à ce que certain seuil est atteint, ils libèrent l'énergie sous forme de lumière. Le phénomène se répète de façon cyclique. L'algorithme de lucioles développé par [38] est inspiré par l'atténuation de la lumière sur la distance et l'attraction mutuelle mais il considère toutes les lucioles comme unisexes.

2.4.2.2 Les lucioles naturelles

Il y a près de deux mille espèces connues de lucioles dans nature, dont la plupart émettent des éclairs de lumière avec un certain afin d'attirer un partenaire ou un appât. En outre pour ces raisons, les lucioles peuvent se protéger contre les attaquants utilisant les flashes qui peuvent également attirer le contraire sexe. La distance entre les lucioles et l'environnement, où la lumière est émise, est en quelque sorte efficace sur l'intensité de la lumière reçue par les lucioles. Comme l'intensité de la lumière obéit à la loi carrée inverse à une distance particulière ($I \propto 1/r^2$), et parce que la lumière est absorbée par l'air, la plupart des lucioles peuvent juste être visible à une distance limitée.



FIGURE 2.7 – Une luciole naturelle

2.4.2.3 Définition (Algorithme firefly)

L'algorithme FireFly (AF) a été développé pour la première fois par Xin-She Yang fin 2007 et 2008 à l'Université de Cambridge[39], qui était basé sur les modèles de clignotement et le comportement des lucioles. La littérature AF s'est considérablement développée au cours des 5 dernières années avec plusieurs centaines d'articles publiés sur les algorithmes de luciole, et Fister et al. ont fourni un examen complet[40], En substance, AF utilise les trois règles idéalisées suivantes :

- 1) Les lucioles sont unisexes de sorte qu'une luciole sera attirée par d'autres lucioles quel que soit leur sexe.
- 2) L'attractivité est proportionnelle à la luminosité, et ils diminuent tous les deux à mesure que leur distance augmente. Ainsi, pour n'importe quelles deux lucioles clignotantes, la moins brillante se déplacera vers la plus brillante. S'il n'y en a pas de plus brillant qu'une luciole en particulier, elle se déplacera au hasard.
- 3) La luminosité d'une luciole est déterminée par le paysage de la fonction objectif.

structure de l'algorithme FireFly

Initialisation

u_i :i Lucioles, $i \in [1, n]$
 n :le nombre de Lucioles
 MaxGeneration : le maximum nombre de génération
 γ : coefficient d'absorption
 r :la distance entre deux firefly
 d :la dimension

2.4.2.4 Caractéristiques de l'algorithme Firefly

Afin de concevoir correctement l'AF, deux problèmes importants doivent être définis : la variation de l'intensité lumineuse et la formulation de l'attractivité. Ces deux problèmes permettent aux développeurs d'adapter différents algorithmes de luciole dans un tel façon qu'ils soient les mieux adaptés aux exigences des problèmes à résoudre. Dans l'algorithme

Algorithme 6 : l'algorithme Firefly**Début**

Définir une fonction objective $f(x)$, $x = (x_1, x_2, \dots, x_d)^T$

Générer une population de Lucioles $x_i (i = 1, 2, \dots, n)$

Définir l'intensité de lumière I à un point x_i par la fonction objective $f(x_i)$

Déterminer le coefficient d'absorption γ

tant que $t < MaxGeneration$ **faire**

Pour $i=1$ jusqu'à n

Pour $j=1$ jusqu'à n

Si $(I_i < I_j)$

 Déplacer la lucioles i vers lucioles j

Fin si

 Varier l'attraction en fonction de la distance r via $\exp[-\gamma r^2]$

 Évaluation des nouvelles solutions et mettre à jour l'intensité de lumière

Fin Pour j

Fin Pour i

 Classer les lucioles et trouver la meilleure solution

fin

Fin

de luciole standard, l'intensité lumineuse I d'une luciole représentant la solution s est proportionnel à la valeur de la fonction de fitness $I(s) \propto f(s)$, tandis que l'intensité lumineuse $I(r)$ [41] varie selon les equations

$$I(r) = I_0 e^{-\gamma r^2}$$

Lorsque I_0 désigne l'intensité lumineuse de la source et l'absorption lumineuse est approximée à l'aide du coefficient d'absorption lumineuse fixe γ . La singularité en $r = 0$ dans l'expression I/r^2 est évitée en combinant les effets de la loi du carré inverse et une approximation de l'absorption sous forme gaussienne. L'attractivité β des lucioles est proportionnelle à leurs intensités lumineuses $I(r)$. Par conséquent, une équation similaire à peut être définie, afin de décrire l'attractivité β [?]:

$$\beta = \beta_0 \exp^{-\gamma r^2}$$

ou β_0 est l'attractivité à $r = 0$. L'intensité lumineuse I et l'attractivité β sont en quelque sorte synonymes. Alors que l'intensité est appelée comme mesure absolue de la lumière émise par la luciole, l'attractivité est un mesure relative de la lumière qui doit être vue dans les yeux des spectateurs et jugé par d'autres lucioles (Yang, 2008).

La distance entre deux lucioles s_i et s_j s'exprime sous la forme euclidienne distance par l'algorithme Firefly de base, comme suit :

$$r_{ij} = \|s_i - s_j\| = \sqrt{\sum_{k=1}^{k=n} (s_{ik} - s_{jk})^2} \quad (2.1)$$

ou n désigne la dimensionnalité du problème. Le mouvement de la i -ème luciole est attirée par une autre luciole plus attrayante j . De cette façon,

$$s_i = s_i + \left\{ \beta 0 e^{-\gamma r_{ij}^2} \right\} (s_j - s_i) + \alpha \in i \quad (2.2)$$

ou $\alpha \in i$ est un nombre aléatoire tiré de la distribution gaussienne. Les mouvements des lucioles se composent de trois termes : la position actuelle de la i -ème luciole[42], attirance pour une autre luciole plus attirante, et une marche aléatoire qui consiste d'un paramètre de randomisation α et du nombre aléatoire généré à partir de l'intervalle $[0, 1]$. Lorsque $\beta 0 = 0$, le mouvement dépend uniquement de la marche aléatoire. Par contre, le paramètre γ a un impact crucial sur la convergence la rapidité. Bien que la valeur de ce paramètre puisse théoriquement capturer valeur de l'intervalle $\gamma \in [0, \infty[$, son réglage dépend du problème à résoudre optimisé. En règle générale, il varie de 0,1 à 10.

En résumé, FA est contrôlé par trois paramètres : le paramètre de randomisation α , l'attractivité β et le coefficient d'absorption γ . Selon au paramétrage, FA distingue deux γ comportements asymptotiques. le premier apparaît lorsque $\gamma \rightarrow 0$ et le second lorsque $\gamma \rightarrow \infty$. Si $\gamma \rightarrow 0$, le l'attractivité devient $\beta = \beta 0$. Autrement dit, l'attractivité est constante partout dans l'espace de recherche. Ce comportement est un cas particulier de particules optimisation en essaim (PSO). Si $\gamma \rightarrow \infty$, le second terme tombe du Éq.(2.2), et le mouvement de la luciole devient une marche aléatoire, qui est essentiellement une version parallèle du recuit simulé. En effet, chaque implémentation de FA peut se situer entre ces deux comportements asymptotiques.

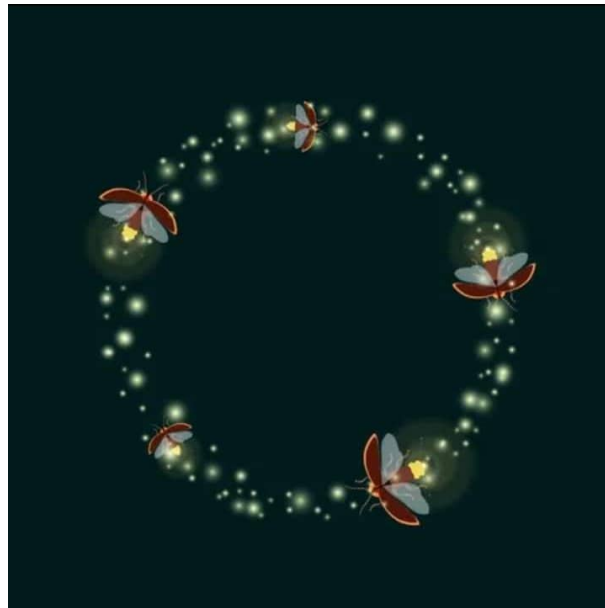


FIGURE 2.8 – Une vue conceptuelle des relations de l'algorithme FireFly

2.5 Conclusion

Dans ce chapitre nous avons présenter un état de l'art quelques méta-heuristiques Qui devise en deux partie Les méta heuristiques à base de solution unique et Les méta heuristiques à base de population de solutions.

Dans ce chapitre nous avons présenté un état de l'Art quelque Méta heuristique qui di- vise en trois parties :les Méta heuristique a base de solution unique qui comprend de quatre

titres la recherche locale simple la recuit simule ,la recherche tabou et la recherche a voisinage variable ,ensuite les Méta heuristique a base de population de solution qui comprend de deux titre l'algorithme génétique et algorithmes évolutionnaires ,en fin les algorithmes a base de l'intelligence par essaim qui comprend de deux titre l'intelligence par essaim de particule et l'algorithme FireFly .

Optimisation multi-objectif

3.1 Introduction

L'optimisation multi-objectif est un axe de recherche très important à cause de la nature multi-objectif de la plupart des problèmes réels. Les premiers travaux menés sur les problèmes multi-objectifs furent réalisés au 19^{me} siècle sur des études en économie par Edgeworth et généralisés par Pareto.

Ce chapitre est consacré à la présentation des concepts de base de l'optimisation multi-objectif. Nous étudierons les principales approches à base de transformation telles que les méthodes d'agrégation des objectifs, méthodes -contrainte et programmation par but ou le but à atteindre

3.2 Problème d'optimisation multi objectif

On dit qu'un problème est multi-objectif s'il comporte plusieurs [43] objectifs qui doivent être optimisés simultanément. Mathématiquement, un problème multi-objectif peut être formalisé comme suit :

$$\min F(x) \quad o F = (f_0, f_1, \dots, f_m) \text{ tel que } G(x) \leq 0 \quad (3.1)$$

Avec $x \in \mathcal{R}^n$, $F(x) \in \mathcal{R}^m$, $G(x) \in \mathcal{R}^p$, on a m fonctions à optimiser ($m \geq 2$) et p contraintes à satisfaire. x appartient à un ensemble des solutions réalisables, et à chaque solution x dans l'espace de décision est associé un vecteur objectif z dans l'espace des objectifs tel que $z = F(x) = (f_0(x), f_1(x), \dots, f_m(x))$ (Voir Figure)

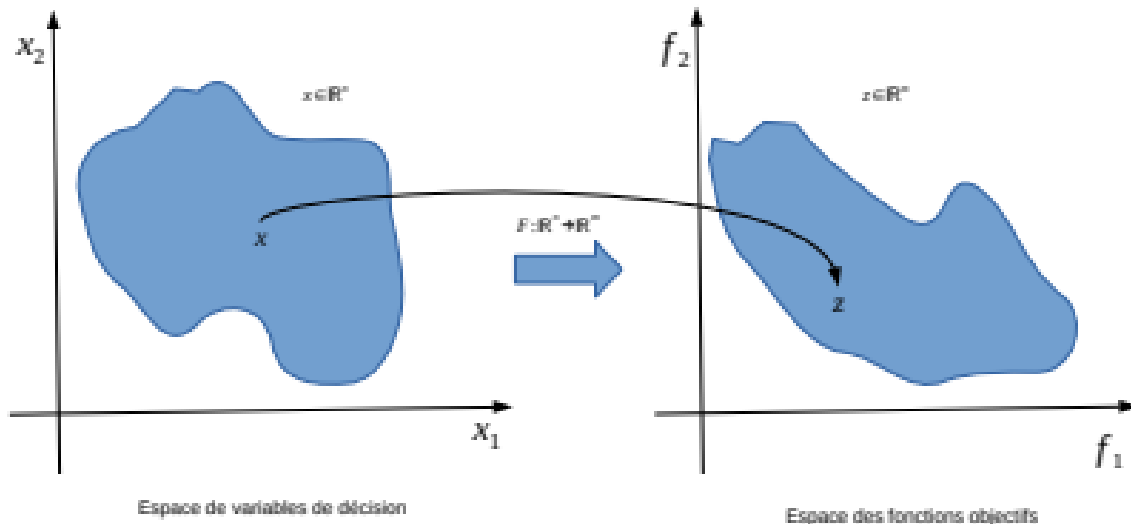


FIGURE 3.1 – Espace de décision et espace objectif d’un problème d’optimisation multi-objectif.

3.2.1 Exemple

Dans le cas de deux objectifs à minimiser, toute amélioration de l’un des objectifs se fait au détriment de l’autre et que la solution optimale ou proche de l’optimum est un compromis entre les deux. Dans l’achat d’une voiture d’occasion, la voiture idéale est celle qui est peu chère (critère économique) avec peu de kilomètres (critère qualitatif), il n’est pas évident de pouvoir regrouper en un seul objectif ces deux critères non commensurables. Ainsi il n’existe plus une solution optimale unique mais un ensemble de solutions. Nous allons donc devoir identifier les meilleurs compromis possibles suivant notre budget.

Les problèmes multi-objectifs ont la particularité d’être beaucoup plus difficiles à traiter que leur équivalent mono-objectif. La difficulté réside dans l’absence d’une relation d’ordre total entre les solutions. Une solution peut être meilleure qu’une autre sur certains objectifs et moins bonne sur les autres. Donc il n’existe généralement pas une solution unique qui procure simultanément la solution optimale pour l’ensemble des objectifs. Voilà pourquoi le concept de solution optimale devient moins pertinent en optimisation multi-objectif.

Dans ce cas la solution optimale ou de bonne qualité n’est plus une solution unique mais, un ensemble de solutions compromis entre les différents objectifs à optimiser. Il est vital pour identifier ces meilleurs compromis de définir une relation d’ordre entre ces éléments. La plus célèbre et la plus utilisée est **la relation de dominance** au sens **Pareto**. L’ensemble des meilleurs compromis est appelé le front Pareto, la surface de compromis ou l’ensemble des solutions efficaces. Cet ensemble de solutions constitue un équilibre, dans le sens qu’aucune amélioration ne peut être faite sur un objectif sans dégradation d’au moins un autre objectif. La solution Pareto consiste à obtenir le front de Pareto PO ou d’approximer la frontière de Pareto PO^* .

3.2.2 Notion de base et quelque définitions

On considère ici le cas de maximisation des objectifs. La minimisation est définie de manière analogue.

3.2.2.1 dominance au sens Pareto

Définition 1 : Soient deux vecteurs objectifs $Y^1, Y^2 \in \Psi / Y^1 = F(S^1)$ et $Y^2 = F(S^2)$. On dit que la solution S^1 **domine** S^2 (Y^1 **domine** Y^2) si et seulement si : $Y^1 \geq Y^2$ et $Y^1 \neq Y^2$ (ie, $Y_k^1 \geq Y_k^2$ pour tout $k = 1, 2, \dots, p$, et $Y_k^1 > Y_k^2$ pour au moins un k). On notera alors $S^1 \succ S^2$. Si S^1 est meilleur que S^2 sur tous les objectifs (ie, $Y_k^1 > Y_k^2$ pour tout $k = (1, 2, \dots, p)$) alors on dit que S^1 **domine fortement** S^2 ; On notera alors $S^1 \succ\!\succ S^2$. Lorsque ni $S^1 \succ S^2$, ni $S^2 \succ S^1$, alors on dit qu'elles sont **incomparables** ou Pareto équivalentes, $S^1 \sim S^2$. La relation de dominance est une relation d'ordre partiel stricte transitive, non réflexive et non antisymétrique [Dupas 2004]. Le point noir est :

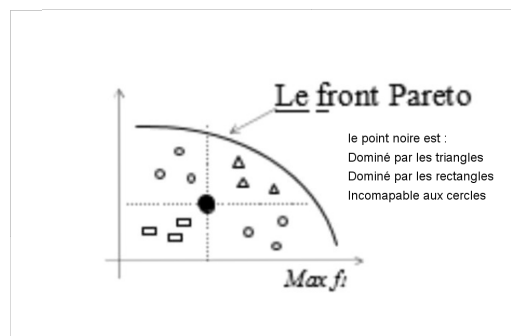


FIGURE 3.2 – Relation de dominance (Cas de deux objectifs à maximiser)

Définition 2 : Une solution est dite Pareto optimale si elle n'est dominée par aucune autre solution réalisable.

3.2.2.2 front Pareto

L'ensemble Pareto optimal

$$PO^* = \{S \in \Omega \Delta \exists S' \in \Omega, F(S') \succ F(S)\} \quad (3.2)$$

L'image de l'ensemble Pareto optimal $F(PO)$ dans l'espace objectif ψ est appelée **frontière Pareto**, ou surface de compromis. L'allure de cette frontière prend des formes différentes selon que les objectifs doivent être minimisés ou maximisés, () cas de deux objectifs.

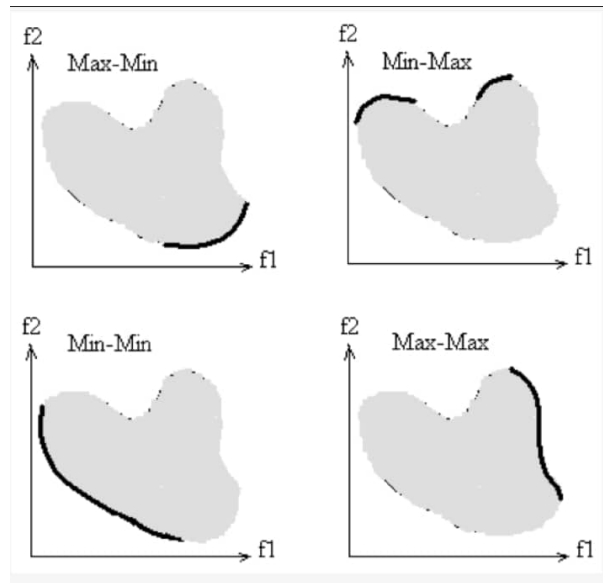


FIGURE 3.3 – Front de Pareto pour un espace à deux dimensions

L'ensemble Pareto optimal regroupe des solutions dites **supportées** correspondants aux sommets de la fermeture convexe de la frontière et des solutions **non-supportées** n'appartenant pas à cette fermeture convexe.

3.2.2.3 convexité

L'ensemble Ψ est dit convexe si tout segment joignant deux points quelconques de ψ est inclus dans ψ . $y_1 \in \Psi \wedge y_2 \in \Psi \Leftrightarrow \text{segment}(y_1, y_2) \subset \Psi$

La convexité est le premier indicateur de la difficulté du problème. En effet, certaines méthodes sont dans l'incapacité de résoudre des problèmes non convexes de manière optimale. Mais il existe d'autres indicateurs tout aussi importants, notamment la continuité, la multi modalité, la nature des variables de décision (entières ou réelles),...

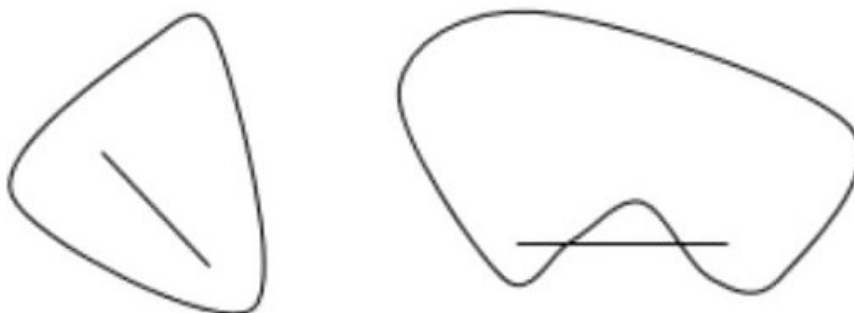


FIGURE 3.4 – espace convexe à (gauche), non convexe à (droite)

3.2.2.4 le point idéal

Les coordonnées du point idéal correspondent aux meilleures valeurs de chaque objectif des points du front Pareto. Les coordonnées de ce point correspondent aussi aux valeurs obtenues en optimisant chaque fonction objectif séparément.

Dans ψ c'est le point de coordonnées (y_1^*, \dots, y_p^*) , avec $y_k^* = \max f_k(S), S \in \Omega$ et $k = 1, 2, \dots, p$. Ce point ne correspond pas à une solution réalisable car si c'était le cas, cela sous-entendrait que les objectifs ne sont pas contradictoires et qu'une solution optimisant un objectif, optimise simultanément tous les autres, ce qui ramènerait le problème à un problème ayant une seule solution Pareto optimale. Une visualisation de l'ensemble de ces définitions est donnée sur la figure 3.5

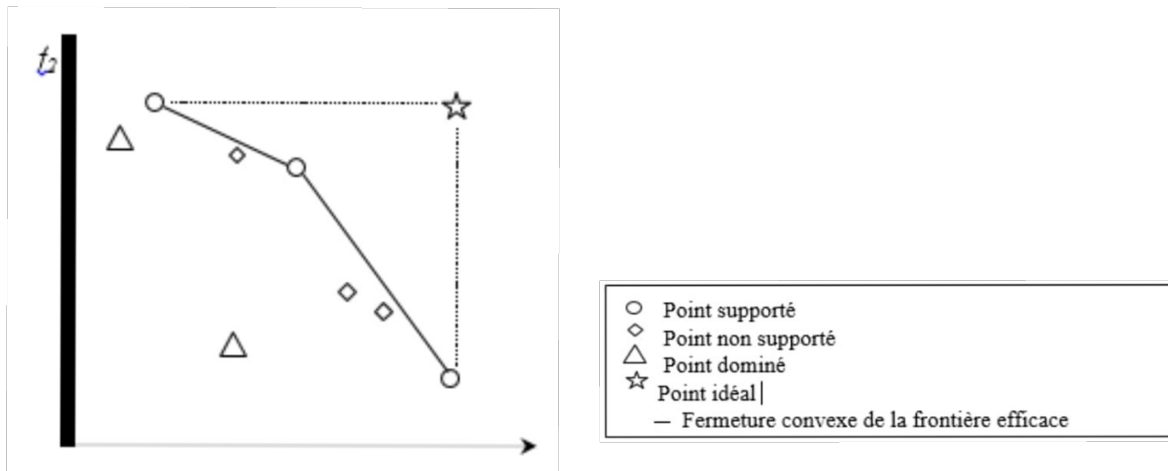


FIGURE 3.5 – Points caractéristiques d'un problème de maximisation bi objectif

3.2.2.5 le point Nadir

correspondent aux pires valeurs obtenues par chaque fonction objectif sur le front de pareto

3.2.2.6 espace de décision

L'espace de décision (vecteur de décision) X de dimension n qui désigne l'ensemble des solutions $x = [x_1, x_2, \dots, x_n]$.

3.2.2.7 espace objectif de dimension

L'espace objectif de dimension (vecteur des critères à optimiser) k qui désigne l'ensemble des évaluations possibles correspondant aux vecteurs objectifs $f(x)$.

3.3 Approches de résolution multi objectif

La résolution de problèmes multi-objectifs relève de deux disciplines assez différentes. En effet, résoudre un problème multi-objectif peut être divisé en deux phases[44] :

3.3.1 La recherche des solutions de meilleur compromis

C'est la phase d'optimisation multi-objectif.

3.3.2 Le choix de la solution à retenir

C'est la tâche du décideur qui, parmi l'ensemble des solutions de compromis, doit extraire celle(s) qu'il utilisera. On parle alors ici de décision multi-objectif et cela fait appel à la théorie de la décision.

Dans le cadre de ce mémoire nous ne parlerons que de la première phase qui consiste en la recherche des solutions de meilleurs compromis. Dans les différentes publications, nous rencontrons deux classifications différentes des approches de résolution de problèmes multi-objectifs. Le premier classement adopte **un point de vue décideur**, les approches sont classées en fonction de l'usage que l'on désire en faire. Le deuxième classement adopte **un point de vue concepteur**, les approches sont triées de leur façon de traiter les fonctions objectives. Ainsi avant de se lancer dans la résolution d'un problème multi-objectif, il faut se poser la question du type d'approche de résolution à utiliser.

3.3.2.1 Classification « point de vue décideur »

On distingue à cet égard trois schémas possibles. Soit le décideur intervient dès le début de la définition du problème, en exprimant ses préférences, afin de transformer un problème multi-objectif en un problème simple objectif. Soit le décideur effectue son choix dans l'ensemble des solutions proposées par le solveur multi-objectif :

1) les approches a priori : le décideur intervient en aval du processus^[45] d'optimisation, pour définir la fonction d'agrégation modélisant le compromis que l'on désire faire entre les différents objectifs. Dans ce cas le décideur est supposé connaître a priori le poids de chaque objectif afin de les mélanger dans une fonction unique. Cela revient à résoudre un problème mono-objectif. Cependant dans la plupart des cas, le décideur ne peut pas exprimer clairement sa fonction d'utilité, parce que les différents objectifs sont non commensurables (exprimés dans des unités différentes).

2) les approches interactives : combinent de manière cyclique et incrémentale les processus de décision et d'optimisation. le décideur intervient de manière à modifier certaines variables ou contraintes afin de diriger le processus d'optimisation. Le décideur modifie ainsi interactivement le compromis entre ses préférences et les résultats. Cette approche permet donc de bien prendre en compte les préférences du décideur, mais nécessite sa présence tout au long du processus de recherche.

3) les approches a posteriori : cherche à fournir au décideur un ensemble de bonnes solutions bien réparties. Il peut ensuite, au regard de l'ensemble des solutions, sélectionner celle qui lui semble la plus appro-

priée. Ainsi, il n'est plus nécessaire de modéliser les préférences du décideur (ce qui peut s'avérer être très difficile), mais il faut en contrepartie fournir un ensemble de solutions bien réparties, ce qui peut également être difficile et requérir un temps de calcul important (mais ne nécessite pas la présence du décideur).

Nous nous placerons dans le cadre de cette troisième famille de méthodes où la modélisation des préférences n'est pas requise et où le procédé d'optimisation doit être puissant afin de fournir une très bonne approximation de la frontière Pareto.

3.3.2.2 Classification « point de vue concepteur »

Ce classement adopte un point de vue plus théorique articulé autour des notions d'agrégation et d'optimum Pareto [46]. Les approches utilisées pour la résolution de problèmes multi-objectifs peuvent être classées en deux catégories [Barichard, 2003] : les approches non Pareto et les approches Pareto (figure 3.6) [47].

1) **Les approches non Pareto** ne traitent pas le problème comme un véritable problème multi-objectif. Elles cherchent à ramener le problème initial à un ou plusieurs problèmes mono-objectifs.

2) **Les approches Pareto** ne transforment pas les objectifs du problème, ceux-ci sont traités sans aucune distinction pendant la résolution.

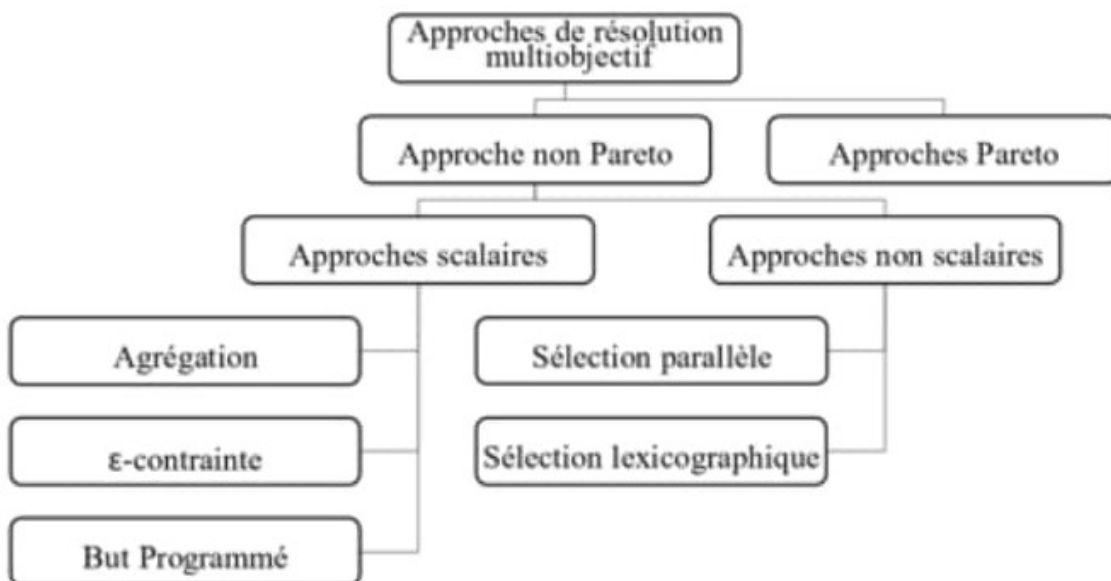


FIGURE 3.6 – Classification « point de vue concepteur »

3.4 Approches non Pareto

Les approches non Pareto sont classées en deux catégories : **les approches scalaires**, qui transforment le problème multi-objectif en problème mono-objectif et **les approches**

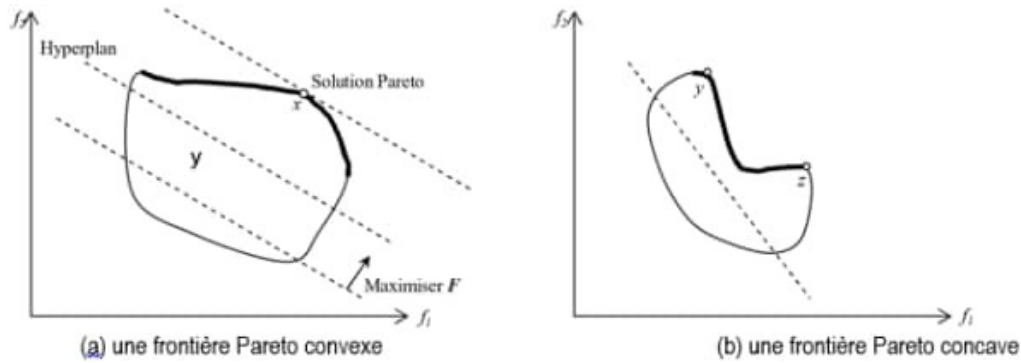


FIGURE 3.7 – l'approche d'agrégation

non scalaires, qui gardent l'approche multi-objectif, mais en traitant séparément chacun des objectifs.

3.4.1 Les approches scalaires « ces approches sont de type a priori »

À l'origine, les problèmes multi-objectifs étaient transformés en problèmes mono-objectifs. Plusieurs approches différentes ont été mises au point pour transformer les problèmes multi-objectifs en problèmes mono-objectifs : les approches agrégées, programmation par but, et les approches ϵ -contraintes, etc.

3.4.1.1 Approche d'agrégation

C'est l'une des premières approches utilisée pour résoudre les problèmes multi-objectifs [Ishibuchi, 1998]. Elle consiste à transformer un problème multi-objectif en un problème mono-objectif, en définissant une fonction objectif unique F comme étant la somme pondérée des différentes fonctions objectifs du problème initial. En affectant à chaque objectif un coefficient de poids qui représente l'importance relative que le décideur attribue à l'objectif :

$$F(s) = \sum_{i=1}^p l_i f_i(s) \quad \text{o les poids } l_i \in [0..1] \quad \text{avec} \quad \sum_{i=1}^p l_i = 1 \quad (3.3)$$

La figure 3.7 illustre le fonctionnement de la méthode d'agrégation. Fixer un vecteur poids revient à trouver un hyperplan dans l'espace objectif (une droite pour un problème bi-objectif) avec une orientation fixée. La solution Pareto optimale est le point où l'hyperplan possède une tangente commune avec l'espace réalisable (point x dans la figure 3.7). Donc, pour une agrégation donnée, il n'existe en général qu'une seule valeur optimale pour le problème. Ces approches résolvent le problème en utilisant différentes valeurs pour les poids qui fournissent différentes solutions supportées. Mais, dans le cas d'une frontière Pareto concave, les solutions non supportées sont alors négligées. La figure 3.7 représente un cas où seules deux solutions Pareto optimales peuvent être trouvées.

Cette approche a l'avantage de pouvoir réutiliser tous les algorithmes classiques dédiés aux problèmes d'optimisation à un seul objectif. Cependant cette approche a aussi deux inconvénients importants. Le premier est dû au fait que pour avoir un ensemble de points bien repartis sur le front Pareto, les différentes valeurs T_i doivent être choisis judicieusement. Il est donc nécessaire d'avoir une bonne connaissance du problème. Le deuxième inconvénient provient du fait que cette méthode ne permet pas, de calculer intégralement la surface de compromis lorsque celle-ci n'est pas convexe[48]

3.4.1.2 But programmé

Dans les approches de ce type, le décideur doit définir des buts T_i ou références qu'il désire atteindre pour chaque objectif f_i . Ces valeurs sont introduites dans la formulation du problème, le transformant en un problème mono-objectif. La nouvelle fonction objectif est modifiée de façon à minimiser les écarts entre les résultats et les buts à atteindre.

$$\min \sum_{i=1}^p |f_i(s) - T_i| \quad \text{avec } s \in \Omega \quad (3.4)$$

Différentes approches sont envisageables, comme celles du min-max [Coello, 1998], ou du but à atteindre. Ces approches, bien que travaillant par agrégation des objectifs, permettent de générer des solutions non-supportées.

3.4.1.3 Approches ϵ -contraintes

Dans cette approche, le problème consiste à optimiser une [49] seule fonction objectif f_k sujette à des contraintes sur les autres fonctions objectif (Convertir $p-1$ des p objectifs du problème en contraintes).

3.4.1.4 Conclusion sur Les approches scalaires

Ces différentes approches de résolution transforment un problème d'optimisation multi-objectif en un ou plusieurs problèmes à un seul objectif. Que ce soit sous la forme d'une somme pondérée, ou sous la forme d'une distance à un but, cette transformation permet d'utiliser facilement les méthodes d'optimisation issues de l'optimisation à un objectif. Mais ces méthodes ont aussi des inconvénients. Certaines ne peuvent traiter complètement des problèmes non convexes et sont donc très sensibles à la forme du front Pareto. Les autres, bien que pouvant traiter les problèmes non convexes, restent quand même sensibles à la forme du front Pareto. Un autre inconvénient important est qu'il faille relancer plusieurs fois les algorithmes de résolution avec des valeurs différentes pour certains paramètres (vecteur de poids par exemple) pour obtenir plusieurs points distincts de la surface de compromis. Ces méthodes nécessitent aussi souvent une bonne connaissance du problème a priori, notamment pour fixer les vecteurs de poids ou les points de référence. Nous présentons dans les sections suivantes comment ces méthodes surmontent les difficultés présentées précédemment.

3.4.2 Les approches non scalaires non Pareto « ces approches sont de type a posteriori »

Ces approches ne transforment pas le problème multi-objectif en un problème mono-objectif, mais utilisent des opérateurs qui traitent séparément les différents objectifs, elles n'utilisent pas non plus la notion de dominance Pareto : sélection parallèle, sélection lexicographique

3.4.2.1 Sélection parallèle

Cette approche a été la première proposant un algorithme génétique pour la résolution de problèmes multi-objectifs [Schaffer, 1984]. L'algorithme proposé, VEGA (Vector Evaluated Genetic Algorithm), sélectionne les individus selon chaque objectif de manière indépendante. L'idée est simple : Pour k objectifs et une population de n individus, une sélection de n/k meilleurs individus est effectuée pour chaque objectif. Ainsi k sous-populations vont être créées et ensuite mélangées afin d'obtenir une nouvelle population de taille n . Le processus se termine par l'application des opérateurs génétiques (croisement et mutation).

3.4.2.2 Sélection lexicographique

Cette approche, proposée par Fourman [Fourman, 1985], elles classent les objectifs en fonction d'un ordre d'importance proposé par le décideur. Ensuite l'optimum est obtenu en optimisant tout d'abord la fonction objectif la plus importante puis la deuxième en intégrant les valeurs obtenues comme contraintes pour la résolution sur des objectifs moins prioritaire et ainsi de suite. La solution obtenue à l'étape k sera la solution du problème. Le risque essentiel de cette méthode est la grande importance attribuée aux objectifs classés en premier. La meilleure solution trouvée pour l'objectif le plus important va faire converger l'algorithme vers une zone restreinte de l'espace d'état et enfermer les points dans une niche.

3.4.2.3 Conclusion sur Les approches non scalaires

Ces différentes approches surmontent les difficultés des approches scalaires. Une seule résolution du problème permet de trouver un ensemble de solutions Pareto optimales. Le décideur peut ainsi choisir une solution suivant la situation courante. L'inconvénient de ces approches est qu'elles tendent à générer des solutions qui sont largement optimisées pour certains objectifs et très peu pour les autres. Les solutions de compromis sont négligées, et ainsi l'aspect multi-objectif du problème est contourné [Meunier, 2002].

Nous présentons dans les sections suivantes l'approche Pareto traitant les problèmes multi-objectifs sans transformation, sans favoriser un objectif par rapport à un autre et fournissant au décideur un ensemble compromis de solutions (supportées et non-supportées) en une seule résolution du problème

3.5 Approches Pareto

« ces approches sont de type a posteriori »

Au 19^{me} Siècle, Vilfredo Pareto, un mathématicien Italien, formule le concept suivant :[50] dans un problème multi-objectif, il existe un équilibre tel que l'on ne peut pas améliorer un objectif sans détériorer au moins un des autres objectifs. Les approches Pareto utilisent directement la notion de dominance dans la sélection des solutions générées.[51] Le principal avantage de ces approches, c'est l'optimisation simultanée d'objectifs contradictoires.

3.6 Conclusion

Nous avons essayé de présenter dans ce chapitre les principaux concepts de l'optimisation multi-objectif, nous avons commencé à mentionner d'une définition complète au problème d'optimisation multi-objectif, ensuite, nous avons discuté du concept de dominance et optimalité de Pareto. Tout en montrant les deux concepts : Front de Pareto et surface de compromis, compte tenu de l'existence d'autres concepts liée à ces deux, nous devons les traiter, tels que le point idéal, point nadir et la convexité. Enfin, Nous avons parlé sur les méthodes utilisées dans le domaine de l'optimisation multi objectifs, nous avons discuté seulement les méthodes à base de transformation, parmi ces méthodes, nous avons consacré de mentionner les méthodes les plus utilisées aux chercheurs, tel que les méthodes d'agrégation des objectifs, méthodes ϵ - contrainte et le but à atteindre. Où nous avons expliqué le concept de base de chaque méthode, et ses avantages et les inconvénients les plus importants.

FA pour la sélection de portefeuille

4.1 Introduction

Dans ce chapitre, nous allons présenter la méthode de résolution de FireFly à notre problème, et l'implémenter par la suite en utilisant le langage de programmation MATLAB, et en dernier on discute les résultats avec un exemple d'application.

4.2 sélection de portefeuille

En sélection de portefeuille, le gestionnaire considère simultanément plusieurs objectifs tels que le rendement, liquidité et le risque de portefeuille. Ces derniers sont conflictuels et incommensurables. De plus, ils peuvent être entachés d'imprécision. Généralement, les gestionnaire de portefeuille cherche la meilleure combinaison de titres qui répond à ses objectifs d'investissement. La modèle du Goal Programming imprécis sera utilisé pour trouver le portefeuille le plus satisfaisant. Les préférences du gestionnaire de portefeuille seront prises en considération, de manière explicite, et ce à l'aide des fonctions de satisfaction

$$\min \lambda \left[\sum_{i=1}^N \sum_{j=1}^N \omega_i \omega_j \sigma_{ij} \right] + (1 - \lambda) \left[- \sum_{i=1}^N \omega_i \mu_i \right] \quad (4.1)$$

$$\text{subject } \sum_{i=1}^N \omega_i = 1 \quad (4.2)$$

$$\sum_{i=1}^N s_i = K \quad (4.3)$$

$$\epsilon_i s_i \leq \omega_i \leq \delta_i s_i, \quad i = 1, \dots, N \quad (4.4)$$

$$s_i \in \{0, 1\}, \quad i = 1, \dots, N \quad (4.5)$$

où k est le nombre souhaité d'actifs investis dans le ou non, si s_i est égal à un, l'actif est choisi pour être investi et la proportion de capital ω_i se trouve dans le $[\epsilon_i, \delta_i]$, où $0 \leq \epsilon_i \leq \delta_i \leq 1$ sinon, l'actif i n'est pas investi et ω_i est égal à zéro

4.3 Représentation et encodage de la solution

la solution est représentée par un vecteur comme suit : $X = (x_1, x_2, \dots, x_N)$

où $x_i \in [0, 1]; i = 1, \dots, N$

Quant au processus de décodage, une sélection est convertie en une sélection d'actifs " s_i " comme dans l'équation suivant :

$$(s_i) \begin{cases} 1 & \text{si } x_i \text{ est sélection} \\ 0 & \text{sinon } i = 1, 2, \dots, N \end{cases} \quad (4.6)$$

et le poids des actifs comme dans l'équation suivant :

$$\omega_i = \frac{s_i x_i}{\sum_{i=1}^n s_i x_i} \quad (4.7)$$

un exemple de décodage est présenté à la table 4.1

0.81	0.91	0.13	0.91	0.63
↓ étape 1				
0	1	0	1	0
↓ étape 2				
0	0.91	0	0.91	0
↓ étape 3				
0	0.5	0	0.5	0

TABLE 4.1 – un exemple de décodage

Dans cet exemple, nous supposons que $N = 5$ et $K = 2$. Le processus de décodage peut être divisé en trois étapes

étape 1 : les valeurs de x_2 et x_4 sont supérieures à celles des autres gènes, de sorte que la sélection de actifs est représentée par $s_2 = 1$ et $s_4 = 1$

étape 2 : puisque la solution est utilisée deux fois, les chaînes, qui interagissent entre elles, sont respectivement $\{0, 1, 0, 1, 0\}$ et $\{0.81, 0.91, 0.13, 0.91, 0.63\}$ jusqu'à peser, le poids avant normalisation est $\{0, 0.91, 0, 0.91, 0\}$

étape 3 : finalement, le portefeuille w est normalisé comme $\{0, 0.5, 0, 0.5, 0\}$

4.4 FireFly algorithm pour PSP

Les principes étapes de la méthode proposée sont présentées dans l'algorithme 1

Algorithme 7 : FireFly pour résoudre les PSP

Data :

N : Nombre des actifs
 μ : Rendement espéré
 N_p : Taille de la population
 λ : coefficient d'absorption de la lumière

Résulte :

la matrice solution x^*

Initialisation :

- 1- Générer une population initiale de lucioles $x_p = x_1, x_2, \dots, x_{N_p}$
- 2- Évaluer la fitness de la population initiale P
- 3- L'intensité lumineuse I_i en x_i est déterminé par $f(x_i)$

Évolutionniste :

Début

tant que $t < T_{max}$ **faire**

pour $i \leftarrow 1$ N_p

pour $j \leftarrow 1$ N_p

Si ($I_i > I_j$)

calculez l'attractivité basée sur l'équation

Déplacez la luciole i vers j en fonction de l'équation

Fin Si

Fin pour j

Fin pour i

fin

Evaleur la population

classez les lucioles, trouvez le meilleur actuel

Fin

4.4.1 Initialisation

Dans le FA , une nombre fini de lucioles sont initialisées aléatoirement dans l'espace de recherche de dimension "d". Les positions initiales des lucioles sont déterminées selon l'équation suivante :

$$x_i = x_{i_{min}} + rand(0, 1)(x_{i_{max}} - x_{i_{min}}), \quad i = \{1, 2, \dots, N_p\} \quad (4.8)$$

où $x_{i_{min}}$ et $x_{i_{max}}$ sont les bornes inférieure et supérieure de la i ème variable, respectivement, et N_p est le nombre total de lucioles.

Nous supposons que :

$$x_{min} = (0, 0, 0, 0, 0) \quad \text{et} \quad x_{max} = (1, 1, 1, 1, 1) \quad \text{et} \quad N_p = 6$$

	position de lucioles x
1	$x_1 = (0.91, 0.21, 0.30, 0.83, 0.40)$
2	$x_2 = (0.71, 0.34, 0.11, 0.01, 0.002)$
3	$x_3 = (0.44, 0.12, 0.33, 0.53, 0.60)$
4	$x_4 = (0.61, 0.56, 0.36, 0.16, 0.22)$
5	$x_5 = (0.22, 0.95, 0.69, 0.55, 0.66)$
6	$x_6 = (0.66, 0.86, 0.78, 0.10, 0.15)$

les propositions des actifs sélectionnés dans chaque solution sont ensuite générées de manière aléatoire à partir des limites inférieure et supérieure données en adoptant une distribution gaussienne . le portefeuille construit de manière aléatoire Pourrait violer les contraintes dans le modèle , il est donc appliqué pou ajuster et normalisé les poids.

	Position	Sélection	Poids
x_1	(0.91, 0.21, 0.30, 0.83,0.40)	$s_1(1, 0, 0, 1, 0)$	$\omega_1(0.52, 0, 0, 0.48, 0)$
x_2	(0.71, 0.34, 0.11, 0.01, 0.002)	$s_2(0, 1, 1, 0, 0)$	$\omega_2(0, 0.76, 0.24, 0, 0)$
x_3	(0.44, 0.12, 0.33, 0.53, 0.60)	$s_3(1, 0, 0, 0, 1)$	$\omega_3(0.42, 0, 0, 0, 0.57)$
x_4	(0.61, 0.56, 0.36, 0.16, 0.22)	$s_4(1, 1, 0, 0, 0)$	$\omega_4(0.52, 0.47, 0, 0, 0)$
x_5	(0.22, 0.95, 0.69, 0.55, 0.66)	$s_5(0, 1, 1, 0, 0)$	$\omega_5(0, 0.57, 0.42, 0, 0)$
x_6	(0.66, 0.86, 0.78, 0.10, 0.15)	$s_6(1, 0, 1, 0, 0)$	$\omega_6(0.45, 0, 54, 0, 0)$

4.4.2 Évaluation

pour différencier les bornes et les mauvais portefeuilles, la fitness d'un portefeuille "p" est évaluée comme suit :

$$f(p) = \lambda \left[\sum_{i=1}^N \sum_{j=1}^N \omega_i \omega_j \sigma_{ij} \right] + (1 - \lambda) \left[- \sum_{i=1}^N \omega_i \mu_i \right] \quad (4.9)$$

où $f(p)$ dénote la fitness de l'individu p , λ désigne la valeur, ω_i et ω_j désignent respectivement la i^{me} et j^{me} dimension du vecteur de proportion chez l'individu p . plus le valeur de fitness et petite, meilleur et le portefeuille.

soit $N = 5$, $\mu = (0.0006218, 0.00027699, 0.00055461, -0.00014033, 0.00069097)$ et

$$\sigma = \begin{pmatrix} 0.00035998 & 0.00011612 & 0.00015121 & 0.00014141 & 0.00012096 \\ 0.00011612 & 0.00015895 & 0.00011577 & 0.00010913 & 0.000938 \\ 0.00015121 & 0.00011577 & 0.00029074 & 0.00012095 & 0.00011799 \\ 0.00014141 & 0.00010913 & 0.00012095 & 0.00024756 & 0.00010714 \\ 0.00012096 & 0.000938 & 0.00011799 & 0.00010714 & 0.00017071 \end{pmatrix}$$

$$f(p) = 0.25 \times \left(\sum_{i=1}^5 \sum_{j=1}^5 \omega_i \omega_j \sigma_{ij} \right) + 0.75 \times \left(- \sum_{i=1}^5 \omega_i \mu_i \right)$$

$$f(p) = 0.25 \times (\omega_1 \omega_1 \sigma_{11} + \omega_1 \omega_2 \sigma_{12} + \dots + \omega_5 \omega_5 \sigma_{55}) + 0.75 \times (-\omega_1 \mu_1 - \omega_2 \mu_2 - \dots - \omega_5 \mu_5)$$

$$f(p) = 0.25 \times (0 + 0 + 0 + 0 + 0 + 0 + 0.5 \times 0.5 \times 0.00015895 + 0.5 \times 0.5 \times 0.00010913$$

$$+ 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0.5 \times 0.5 \times 0.00010913 + 0 + 0.5 \times 0.5 \times 0.00024756$$

$$+ 0 + 0 + 0 + 0 + 0 + 0 + 0.75) \times (-0 - 0.5 \times 0.00027699 - 0 - 0.5 \times (-0.00014033) - 0)$$

$$f(p) = 0.25 \times (0.000156199) + 0.75 \times (-0.0006833)$$

$$f(p) = -0.000012199$$

4.4.3 L'attractivité de FireFly

La FA s'appuie sur l'ensemble initial de lucioles qui communiquent entre elles en émettant une lumière clignotant. Les lucioles se caractérisent par leur attractivité et l'intensité lumineuse émise par elles. même l'intensité lumineuse est associée à la fonction objectif codée. pour un problème de minimisation une luciole de faible intensité attirera une autre luciole de faible intensité.

Supposons que x_i représente une solution pour une luciole i , alors que $f(x_i)$ détermine sa valeur de fitness. Ici, la luminosité I d'une luciole est déterminée comme suite :

$$I_i = f(x_i)$$

4.4.4 Attractiveness calculation

La fonction attractivité B , peut être formulée comme une fonction monotone décroissante sous la forme généralisée suivante :

$$B = \beta_0 e^{-\lambda r^2} \quad (4.10)$$

où r est la distance entre deux lucioles, β_0 est la valeur d'attractivité maximale et est le coefficient d'adsorption de la lumière, la valeur détermine la variation d'attractivité en fonction des distances entre les lucioles communiquées. la distance affectant l'attractivité entre deux lucioles i et j aux positions x_i et x_j , peut être définie comme la distance cartésienne ou euclidienne comme suite :

$$r_{ij} = \|x_i - x_j\| \quad (4.11)$$

4.4.5 Le mouvement towards attractive fire fly

une nouvelle population est générée par l'attractivité des autres lucioles avec une intensité plus élevées. Pour chaque i -ème solution, si existe une x_j tel que $I_j > I_i$. la i -ème solution change de position selon :

$$x_i(t+1) = x_t + \beta_0 e^{-\lambda r^2} (x_j(t) - x_i(t)) + \alpha(\text{rand} - 0.5) \quad (4.12)$$

où rand est un nombre aléatoire généré uniformément distribué entre 0 et 1. Généralement, le paramètre de randomisation α est considéré comme une fonction décroissante avec schéma de réduction à progression géométrique à partir du α_0 initial selon :

$$\alpha = \alpha_0 \theta^t, \quad 0 < \theta < 1 \quad (4.13)$$

4.5 Réalisation

4.5.1 Description du langage

MATLAB est un logiciel de calcul numérique qui permet de réaliser des simulations numériques basées sur des algorithmes d'analyse numérique ou des algorithmes de portefeuille, ce logiciel commercialisé par la société MathWorks.

Il a été initialement développé à la fin des années 70 par Cleve Moler, professeur de mathématique à l'université du Nouveau-Mexique puis à Stanford, pour permettre aux étudiants de travailler à partir d'un outil de programmation de haut niveau et sans apprendre le Fortran ou le C.

On trouve des applications de MATLAB dans de nombreuses disciplines : Il constitue un outil numérique puissant pour la modélisation de systèmes physiques, la simulation de modèles mathématiques, la conception et la validation (tests en simulation et expérimentation) d'applications.

4.5.1.1 Présentation des résultats

L'application de l'algorithme FA sur ce PSP donne les résultats suivants. L'augmentation du coefficient λ fait peu de différence avec les résultats, car la valeur de la fonction augmente à mesure que la valeur du coefficient λ augmente. En fonction de ce résultat, nous concluons que le coefficient λ a un effet négatif sur l'amélioration du problème PSP lorsqu'il est proche de 1.

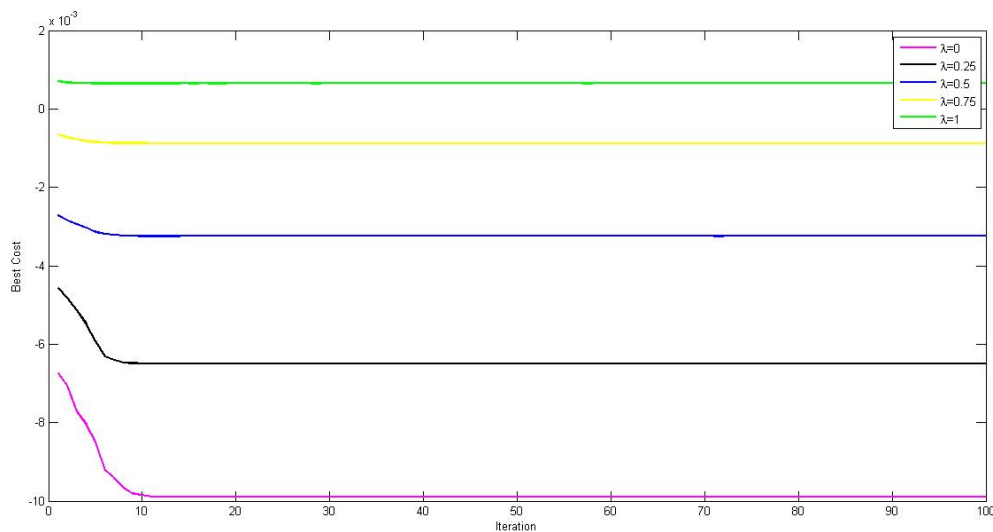


FIGURE 4.1 – Meilleure solution avec des coefficients λ différents à chaque génération

4.5.1.2 Paramètres

Nous avons appliqué l'algorithme FA avec ces paramètres :

- N_p (taille de la population) : 25
- T_{max} (Nombre de générations) : 100
- K (Nombre d'actifs sélectionnés) : 10
- L_b (Borne inférieures) : 0.01
- U_b (Borne supérieures) : 1
- γ (Light absorption coefficient) : 1
- β_0 (Attraction coefficient base value) : 2

4.5.1.3 Source des données

Afin d'évaluer les performances de l'algorithme FireFly, nous avons utilisé l'instance PSP (Hang-Seng) de L'ORLibrary, qui a été générée en collectant des données provenant des principaux marchés financiers. Les auteurs ont utilisé DATASTREAM pour obtenir des données de prix hebdomadaires pendant la période de mars 1992 à septembre 1997. Ils ont collecté 291 valeurs pour chaque stock. L'instance est composée de 31 actifs. Pour chaque actif, un rendement et un écart type estimés sont fournis. Il existe également une matrice de corrélation entre tous les actifs.

Une matrice de covariance entre tous les actifs est formée par la relation suivante :

$$COV = F \times COR \times F^t,$$

où COV est une matrice de covariance, F est un vecteur d'écart type, et COR est une matrice de corrélation

écarttype	Rendement	Actifs
0.043208	0.00139	1
0.004058	0.004177	2
0.041342	0.001487	3
0.44896	0.004515	4
0.069105	0.010865	5
0.053671	0.001759	6
0.046613	0.002594	7
0.045492	0.00495	8
0.0536334	0.007115	9
0.046923	0.003186	10
0.047043	0.002039	11
0.042955	0.005202	12
0.039754	0.004489	13
0.050224	0.003642	14
0.036183	0.00396	15
0.038844	0.000141	16
0.050467	0.000382	17
0.050467	0.000392	18
0.05871	0.005294	19
0.05	0.004801	20
0.044716	0.002699	21
0.038128	0.001879	22
0.045625	0.004656	23
0.050449	0.003842	24
0.05335	0.00269	25
0.04545	0.004793	26
0.47931	0.003286	27
0.036047	0.0023338	28
0.035848	0.005817	29
0.036762	0.001993	30
0.039827	0.00238	31

FIGURE 4.2 – Le rendement espéré et la variance de chaque actif

0.562289	0.716125	0.707857	0.338386	0.704836	0.800165	0.628523	0.379162	0.724315	0.566884	0.638969	0.477964	0.427249	0.471693	0.537818	0.469064	0.659233	0.640393	0.53771	0.676022	0.694944	0.702392	0.306208	0.676865	0.262308	0.541716	0.306844	0.473943
0.562289	0.652125	0.570407	0.405845	0.524819	0.428105	0.56449	0.429524	0.589247	0.680078	0.535283	0.532955	0.429611	0.355394	0.508284	0.531914	0.523236	0.589489	0.581244	0.490808	0.624840	0.549882	0.395293	0.257454	0.554135	0.535374	0.447347	0.447248
0.748125	0.653215	0.757165	0.338075	0.69641	0.272737	0.68274	0.500445	0.745949	0.576614	0.629541	0.528111	0.687167	0.591806	0.482866	0.510442	0.726139	0.685113	0.599002	0.74486	0.798955	0.302451	0.73475	0.287452	0.622941	0.585301	0.627182	0.707857
0.338386	0.458485	0.380073	0.3791	0.341611	0.376146	0.36979	0.316438	0.32337	0.400683	0.489095	0.417346	0.396052	0.33356	0.273437	0.340089	0.555389	0.317963	0.20173	0.411425	0.379793	0.374095	0.714444	0.39173	0.167192	0.239203	0.345611	0.229771
0.704836	0.524819	0.69641	0.711474	0.341611	0.686077	0.657993	0.40358	0.738586	0.505409	0.618204	0.456892	0.632252	0.50067	0.435105	0.523773	0.52415	0.673406	0.671824	0.571706	0.71618	0.708831	0.678212	0.297025	0.674489	0.359757	0.577717	0.544484
0.800165	0.553856	0.72737	0.68274	0.500445	0.745949	0.576614	0.629541	0.528111	0.687167	0.591806	0.482866	0.510442	0.726139	0.685113	0.599002	0.74486	0.801289	0.739597	0.740014	0.721679	0.389855	0.801464	0.37935	0.621502	0.580636	0.53082	0.546794
0.628523	0.379162	0.724315	0.566884	0.638969	0.477964	0.427249	0.471693	0.537818	0.469064	0.659233	0.640393	0.53771	0.676022	0.694944	0.702392	0.306208	0.676865	0.262308	0.541716	0.306844	0.473943	0.537818	0.469064	0.659233	0.640393	0.53771	0.676022
0.379162	0.56449	0.429524	0.589247	0.680078	0.535283	0.532955	0.429611	0.355394	0.508284	0.531914	0.523236	0.589489	0.581244	0.490808	0.624840	0.549882	0.395293	0.257454	0.554135	0.535374	0.447347	0.447248	0.554135	0.535374	0.447347	0.447248	0.554135
0.477964	0.563283	0.528111	0.557505	0.417346	0.456892	0.68606	0.553166	0.407488	0.545683	0.464313	0.517448	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331
0.632612	0.552895	0.877167	0.642025	0.360562	0.632252	0.412139	0.64201	0.456407	0.678357	0.617838	0.52446	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331
0.545067	0.429611	0.391806	0.508915	0.3356	0.46231	0.273437	0.435105	0.506935	0.477371	0.342712	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779
0.427249	0.355394	0.482361	0.273437	0.435105	0.506935	0.477371	0.342712	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779	0.532779
0.471693	0.508369	0.448244	0.506371	0.277162	0.522473	0.589245	0.444542	0.355586	0.505349	0.475298	0.464431	0.333389	0.45292	0.471532	0.328293	0.47331	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592
0.537818	0.563444	0.570296	0.596545	0.340891	0.52415	0.57298	0.557942	0.428565	0.506999	0.513903	0.52057	0.464431	0.52306	0.507084	0.443367	0.47331	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592
0.469064	0.51914	0.519442	0.533082	0.553586	0.501132	0.2979	0.4673	0.375021	0.527463	0.459783	0.607633	0.517448	0.254241	0.496463	0.417895	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592	0.545311
0.659323	0.562836	0.723139	0.801289	0.308989	0.673406	0.26571	0.637523	0.461947	0.724934	0.623799	0.671506	0.539304	0.720173	0.575823	0.48293	0.47331	0.398592	0.545311	0.398592	0.545311	0.488001	0.52446	0.488148	0.483266	0.48263	0.47331	0.398592
0.640393	0.539489	0.695113	0.719058	0.317963	0.671824	0.40939	0.730245	0.475026	0.709387	0.637488	0.57901	0.537215	0.644867	0.579715	0.488054	0.517006	0.390538	0.50864	0.701304	0.616889	0.612202	0.688152	0.278376	0.684355	0.295948	0.245383	0.464195
0.53771	0.581244	0.599002	0.575522	0.270173	0.571706	0.788997	0.582554	0.458584	0.618255	0.603506	0.701701	0.557805	0.482189	0.646984	0.430993	0.37552	0.529696	0.505788	0.614224	0.616889	0.612202	0.688152	0.278376	0.684355	0.295948	0.245383	0.464195
0.676022	0.490808	0.74486	0.82692	0.411425	0.71618	0.780014	0.480293	0.517916	0.709248	0.501547	0.670854	0.540141	0.686433	0.582884	0.477334	0.603157	0.529306	0.246174	0.685987	0.735107	0.612202	0.788289	0.294116	0.721083	0.301172	0.628874	0.556886
0.694944	0.652484	0.789555	0.805143	0.337973	0.708831	0.721679	0.678545	0.447935	0.788883	0.654622	0.64864	0.605197	0.49558	0.597248	0.530744	0.576297	0.518073	0.785987	0.735107	0.612202	0.788289	0.294116	0.721083	0.301172	0.628874	0.556886	0.544335
0.702592	0.394982	0.302451	0.316897	0.174444	0.297025	0.801464	0.304839	0.185522	0.287789	0.321114	0.680335	0.291848	0.544896	0.647036	0.563318	0.48829	0.574881	0.541925	0.259895	0.881532	0.600488	0.742809	0.725457	0.341727	0.724421	0.324742	0.595338
0.306208	0.595283	0.302451	0.316897	0.174444	0.297025	0.801464	0.304839	0.185522	0.287789	0.321114	0.680335	0.291848	0.544896	0.647036	0.563318	0.48829	0.574881	0.541925	0.259895	0.881532	0.600488	0.742809	0.725457	0.341727	0.724421	0.324742	0.595338
0.262308	0.554135	0.287452	0.301167	0.167192	0.350757	0.621502	0.318516	0.247678	0.401257	0.260654	0.576251	0.294688	0.222381	0.340381	0.279603	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381	0.340381
0.541716	0.535374	0.629041	0.579396	0.577717	0.629026	0.559208	0.618103	0.428385	0.638766	0.603616	0.545427	0.727085	0.481897	0.43206	0.476747	0.4285287	0.6025536	0.61254	0.588872	0.626874	0.606545	0.595338	0.319414	0.585657	0.387749	0.1	0.505442
0.506844	0.447247	0.365301	0.545811	0.544484	0.553082	0.560111	0.359789	0.562223	0.514882	0.446152	0.575802	0.571055	0.488616	0.431646	0.58248	0.483781	0.579138	0.646195	0.70653	0.555686	0.652223	0.577495	0.579388	0.280573	0.505442	0.1	0.602896
0.473943	0.447248	0.387182	0.512352	0.229771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234	0.249771	0.75234

FIGURE 4.3 – La matrice de Corrélation

0.001866931264	0.00097803533322896	0.001332806919598	0.00100441120333224	0.00163452636172285	0.00136988938171316	0.00123740195173373	0.000878676795183264	0.00146851171084596
0.00978083533322896	0.001620706564	0.00104057423194074	0.00129599434143105	0.00113396965258164	0.00107953947147312	0.00101434125950522	0.000992436327656106	0.001066363836238166
0.001332806919598	0.00104057423194074	0.001709160964	0.00096585962199825	0.00154524080672962	0.0014024035216561	0.00128312258042321	0.00110965512888846	0.00144317972108003
0.00137314798765338	0.00103096682698938	0.00140536671194528	0.001176172186128	0.0017432896097606	0.00155974636925587	0.00143588693561395	0.00122833728303149	0.00152843719464422
0.0010044112033224	0.00129599434143105	0.00096585962199825	0.004775501025	0.00126701280810701	0.0012116382471929	0.0011562304927014	0.00117283870549566	0.00105504928952355
0.00163452636172285	0.00155974636925587	0.00154524080672962	0.002880576241	0.002880576241	0.00174142199681487	0.0015552698989828	0.00123882441338821	0.00186005818269314
0.00139698938171316	0.00143588693561395	0.0014024035216561	0.0012116382471929	0.00174142199681487	0.002172771769	0.00146062381151778	0.0014265655268431	0.00172653164592023
0.00123740195173373	0.0011562304927014	0.0015552698989828	0.00117283870549566	0.0015552698989828	0.00146062381151778	0.002069522064	0.00127108256434538	0.00137653172868376
0.000878676795183264	0.00152843719464422	0.00123882441338821	0.00105504928952355	0.00123882441338821	0.00114265653268431	0.00127108256434538	0.002876605956	0.0010559084024490
0.00146851171084596	0.00117550795797907	0.00110965512888846	0.00105504928952355	0.00186005818269314	0.00172653172286376	0.00137653172286376	0.0010559084024490	0.002201767929
0.00115228798704994	0.0013573704880236	0.00112142872160348	0.00070489405964745	0.00147806690054988	0.00137588734671206	0.00127563329842661	0.00101680597137748	0.00146633742332499
0.0018698938171316	0.00099503246205792	0.0011796762116501	0.00146611020734888	0.00142523087280222	0.00133205446286413	0.00130636671741378	0.00102237092458969	0.00121926110455033
0.00123740195173373	0.0014927115094656	0.000867955639899348	0.00114653300214882	0.000973134736501928	0.00107167255220626	0.00100401158640869	0.000868832073677568	0.00101417373195739
0.000878676795183264	0.00087870125457072	0.00142680648991754	0.00125141317119024	0.00170428098034861	0.00160823608769152	0.00146752303538861	0.00122942995290051	0.001598699737638
0.000852155057349288	0.000841505220226944	0.00085269330660316	0.000839143037754	0.00097229002162131	0.0010324325269488	0.000945620294552388	0.00089591541089652	0.000976944770232417
0.000717082567220448	0.000671817842258472	0.000791648503249968	0.00073399093674894	0.00090710530455402	0.000896579785206132	0.000843558116149008	0.000713992162508352	0.000971084035776348
0.00089635247211312	0.00062959583302296	0.00081500672564304	0.0008423612548398	0.00123327169588434	0.00103923689595969	0.00088941214312272	0.00083876454906552	0.0010428735976546
0.00117275417194725	0.00103285354438733	0.00118986940336734	0.00118886487403319	0.00141972021522155	0.0013391023386754	0.00128094821931569	0.00116001706148307	0.001303955314231156
0.00118989419938752	0.00133172647728792	0.00126078381503844	0.00225411043927995	0.00157907916465212	0.00147039615798054	0.0013014988051436	0.00118277498967894	0.00145308108214979
0.0014244014092	0.0010428714126	0.0014948006269	0.00106763424225	0.0018071186713	0.0016985544135	0.0014500643238	0.0012388032699	0.0017008039041
0.0012372962248887	0.00101320426888061	0.00128501986336254	0.00098253720557034	0.00161234564536326	0.00151389659317784	0.00148547918252664	0.00113925387914654	0.0014884209424392
0.0008584207167104	0.000828092429920736	0.000944195530399552	0.00071186139533112	0.00116992079977693	0.00108304475415686	0.0010116651308279	0.000707215711704768	0.00110610777442872
0.00133268611003	0.001067612269485	0.001404976346725	0.00129718931101562	0.0017537381655875	0.00157163831422063	0.0016672928602425	0.0016572928602425	0.00151840200312
0.00151483920361805	0.00119991677526034	0.00153703945043089	0.00117826788350009	0.00191926503725185	0.00178723314222932	0.00155268830292786	0.00155268830292786	0.0018674110299785
0.00161912233657856	0.0013416317230812	0.00169356456117205	0.00137880761839163	0.00194141086730042	0.00179416930481205	0.0016161447045438	0.0016161447045438	0.00179212420277127
0.0006052599799488	0.0006401882273202	0.0005683035840489	0.000547897596579	0.00072454702782375	0.0008255078280675	0.0006302885915646	0.0006302885915646	0.0006135269954815
0.00140176849731204	0.00114866176636403	0.0014559537096195	0.00129751621458615	0.0017351518483198	0.00179063704247719	0.00147852915751195	0.00147852915751195	0.00165204716208152
0.000408549635095008	0.000373612128159204	0.000428376801531448	0.00041647492450852	0.000678602039602509	0.000636028672783	0.000523958532219984	0.000523958532219984	0.000678699484159917
0.000839074954738944	0.00079971033412184	0.000921882189100656	0.00064211738139612	0.0011152635718774	0.00103851916708165	0.00100800605160125	0.00100800605160125	0.0010743141558479
0.000805236184372224	0.000792294606574812	0.00089549007535804	0.00087800332107411	0.00107429595408617	0.000994970362879416	0.000936716700076344	0.000936716700076344	0.0009776343000105
0.000815582449418088	0.0007109587737402	0.000668019836583788	0.000632386047982785	0.00101583877342558	0.0010267737029098	0.00094661087466838228	0.00094661087466838228	0.000988355325309591

0.00115228798704694	0.00118592623397116	0.000818933151827648	0.0013728177592423	0.000852155057349288	0.000717082567220448	0.00089635247211312	0.0011725417194725	0.00118989419938752	0.0014244014092
0.000813835845862056	0.00101897446046033	0.0010884080742495	0.0113891191963114	0.00080508605350173	0.000671817842558472	0.00062959563302296	0.001082855354436733	0.00133172647728792	0.0010428714126
0.00112142872160348	0.00011796762116501	0.000867955639899348	0.00142860648991754	0.000885269330660316	0.0007916448503249968	0.00081500672564304	0.00118986940336734	0.00126078381503844	0.0014948006269
0.00117550795797907	0.0013573704880296	0.00099503246205792	0.0014927715094656	0.00087870125457072	0.000841505220226944	0.0009978682195	0.0013516346361	0.0014051210765	0.0017986886512
0.00070489405964745	0.00146611020734888	0.00114653300214882	0.00125141317719024	0.000839143037754	0.00073399099674894	0.0008423612548398	0.00118886487403319	0.00225411043927995	0.001066763424225
0.0014780669005988	0.00142523087280222	0.000973134736501928	0.00170428098034861	0.0009722902162131	0.00090710530455402	0.00123327169588434	0.00157907916463212	0.0018071186713	0.0018071186713
0.00137589734671206	0.00133205446286413	0.00107167255220626	0.001608236808769152	0.00107167255220626	0.00089957972506132	0.0013391023386754	0.00147039615790054	0.0016985544135	0.00151389865931784
0.00127563329842661	0.00130636671741378	0.00100401158640869	0.00146752303538861	0.00100401158640869	0.000843558116149008	0.000843558116149008	0.00088941214312272	0.001280948219315669	0.001301498051436
0.00101680597137748	0.00102297092458869	0.000868832073677568	0.00122942993290051	0.000868832073677568	0.000713992162508352	0.000713992162508352	0.00083876454906552	0.00116001706148307	0.00118277498967894
0.00146633742332499	0.00121926110455033	0.00101417373195739	0.001598699737638	0.00101417373195739	0.000971084035776348	0.00104287535976546	0.00145308109214979	0.0017008039041	0.0017008039041
0.002213043849	0.00110891915604212	0.000867725962038936	0.00145975820117962	0.000867725962038936	0.000973543712475672	0.000973543712475672	0.00098336833893972	0.00122006691808314	0.001187011531676899
0.00110891915604212	0.001845132025	0.00092449034486809	0.00123954176772712	0.00092449034486809	0.00064564477708302	0.0009077342558064	0.00124099317525304	0.00136700707071885	0.00130504377575
0.000867725962038936	0.00092449034486809	0.001580380516	0.000995908518729696	0.001580380516	0.000743098941409968	0.001011152976	0.0014551099643	0.0014411514345	0.0025
0.00145975820117962	0.00123954176772712	0.000995908518729696	0.002522450176	0.000995908518729696	0.000962313160288896	0.00101674900421808	0.00133265669459614	0.0013353205677504	0.0015681096312
0.000973543712475672	0.00064564477708302	0.000743098941409968	0.000962313160288896	0.000743098941409968	0.00067833282210876	0.00106402037113162	0.00112772685860544	0.0011709566366	0.0011709566366
0.00098336833893972	0.0009077342558064	0.00058324064272188	0.0010014330304384	0.00058324064272188	0.00056804234058216	0.0011745627036662	0.00146725774228385	0.0013937144149875	0.0017022094375
0.00122006691808314	0.00124099317525304	0.000931771715017858	0.00145059804008605	0.000931771715017858	0.0008691501305988316	0.0011190820006572	0.00155103799444935	0.00153446010505767	0.00198261290815
0.00118701531678699	0.00136700707071885	0.00120770155766832	0.00155223182912784	0.00120770155766832	0.0009530226025398	0.000496782381339	0.00072854903821905	0.000642964977981	0.00066969666
0.00146726881785	0.00130504377575	0.0010703844008	0.0018084984376	0.0010703844008	0.0014551099643	0.001030032566082402	0.00152498267124425	0.0005260771274	0.00163274315295
0.00134100578802733	0.00128981246092468	0.0009549749953876	0.0014484774573933	0.00093795511791702	0.00101674900421808	0.00050443048359198	0.00044557834543728	0.00090201966361896	0.0010799855264
0.00108248185859702	0.00091372368688424	0.00081517282605616	0.00126805264865101	0.000892307574730656	0.00078725496601088	0.0006701474877024	0.000862502551954952	0.00090201966361896	0.0010645135578
0.00124819577064563	0.00137520896950937	0.00097969491745125	0.00157294062651	0.0009622536870975	0.0009858625243875	0.00069788212768296	0.00108569956895979	0.00104414309063262	0.0010645135578
0.00155359626415295	0.00146676752284293	0.00121375256859056	0.00164581795664541	0.00109021420994842	0.00117145627036662	0.0011190820006572	0.00146725774228385	0.00153446010505767	0.00198261290815
0.00156705506238042	0.001488328439477	0.00115474242974662	0.00179718070328864	0.00113623380971699	0.0011190820006572	0.0011190820006572	0.00155103799444935	0.00173026438425	0.00193578049171472
0.0006865752402459	0.0006623662417515	0.0004789859896107	0.0006689533045248	0.00041014427160735	0.0004738299902622	0.000496782381339	0.00072854903821905	0.000642964977981	0.00066969666
0.000953873624767392	0.00088734058640084	0.000932767338801584	0.00102998896908982	0.000670474803888864	0.000670474803888864	0.000670474803888864	0.000862502551954952	0.00090201966361896	0.0010799855264
0.0006865752402459	0.0006623662417515	0.0004789859896107	0.0006689533045248	0.00041014427160735	0.0004738299902622	0.000496782381339	0.00072854903821905	0.000642964977981	0.00066969666
0.00134331460761385	0.00131425267848518	0.0010384620690341	0.00155760105629358	0.000977302151858214	0.000929582801458976	0.001030032566082402	0.00139084096203324	0.00152498267124425	0.00163274315295
0.000442006371859734	0.000451189711778948	0.000422219918707444	0.00091523029806252	0.00095349291628536	0.000656407148407296	0.00106275393908218	0.00075959420243553	0.000642044062078848	0.00063788212768296
0.000874572884022312	0.00088084405355837	0.00071551705361136	0.000960944309437936	0.000800259715968507	0.000701324268880004	0.00059838392933958	0.000915340919524645	0.000933452345239413887	0.001042232763

0.001237292248687	0.00086584207167104	0.00133286611003	0.00161912233857856	0.00060552599799488	0.00140176849731204	0.00040859635095008	0.000839074954736944	0.000805526194372224	0.00001558249418088
0.00101320428888061	0.0008280952429920736	0.0010676122684835	0.00134168317230812	0.00064004882273202	0.00114866176636403	0.000373612128159204	0.00079971033412184	0.000792294606574812	0.000717095877377402
0.00128501986336254	0.000944199530399552	0.001404976346725	0.00169356456117205	0.000568035840489	0.0014559337096195	0.000428376801531448	0.000921882189100656	0.000889549007535804	0.000868019836583788
0.00144355898354	0.0010226638060	0.0016993816536	0.0017866709038	0.0006450052819	0.0015206250067	0.0004875984668	0.0009624337834	0.0009497775971	0.0009161227735
0.00098253720557034	0.00071186139533112	0.00129718831101583	0.00137880761639163	0.000547897596579	0.00129751621458615	0.00041647994250652	0.00087800632107111	0.00087800632107111	0.000632386047982785
0.00116892079977693	0.00116892079977693	0.001537391655875	0.00194140867390042	0.00072454702782375	0.0017351518483188	0.00067860203960509	0.00111152633718774	0.001074295959408617	0.0010158387342558
0.00108304475415696	0.001571638331422063	0.00178723314222932	0.00082550782900675	0.00082550782900675	0.0017906370427719	0.00063602836727283	0.00103851916708165	0.000994970362879416	0.00102877237029098
0.0014500643238	0.0010116651380279	0.0016672928602425	0.0016161447045438	0.000630859156646	0.00147852915751195	0.0005239835219984	0.00108800665100125	0.000938716700076344	0.000946610874663828
0.0012388032899	0.00113925367914654	0.000707215711704768	0.00126766985195	0.00128726194663878	0.0004522405417866	0.00120349811948512	0.000478846972879044	0.000823065233941168	0.000769113208565292
0.00148844209424392	0.00110610774442872	0.00151840200312	0.00179212420277127	0.00061358269954815	0.00165204716208152	0.000678699484159917	0.0010743141558479	0.0009776343000105	0.000988355353209591
0.00146726881785	0.00134100578802733	0.00108248185859702	0.00155359626412295	0.00156705506238042	0.0008865752402459	0.00134331460761385	0.000442006371659734	0.000953873624767392	0.00091523029806252
0.00128881246092468	0.00091372368688424	0.00137520896950938	0.00146676752294293	0.0006623682417515	0.00131425267848518	0.00045188711778948	0.00088734058640084	0.00095349291628536	0.00088084405355837
0.0015681096312	0.0011709566336	0.0017022094375	0.00198261290815	0.000668969666	0.00163274315295	0.0005200771274	0.0010798855264	0.0010645133578	0.001042322763
0.001998520656	0.00105171948077011	0.0014191629548425	0.00164119287671472	0.0005616906078672	0.00146719481750478	0.00047887079588896	0.00098188885956672	0.00106224738710244	0.00105173964871458
0.00105171948077011	0.001453744384	0.00107005834039	0.0012211265037544	0.000557811195824	0.00110333193761416	0.000337254399126128	0.000802144011322368	0.00098190595581408	0.000906450926838224
0.0014191629548425	0.00107005834039	0.002081640625	0.00180780172337188	0.000609895481625	0.00155378449941	0.0004953208357075	0.00102529317381	0.0009320333733975	0.000982930795240625
0.0016583132963079	0.00117786246779734	0.00176839816159583	0.00195198527459366	0.00062433816900479	0.00174365408530377	0.0005971631196069	0.00123076274154484	0.00104270222672477	0.00103344261763808
0.00164119287671472	0.0012211265037544	0.0012211265037544	0.00284462225	0.00082857213382025	0.00185190976109158	0.00062433816990479	0.00113825832274104	0.00113229517991865	0.0010747274762908
0.0005616906078672	0.0005577811195824	0.000609895481625	0.0006555191965245	0.0020657025	0.00078437980830095	0.00022861178803485	0.0005204185471224	0.0005500131281865	0.0004889252847636
0.00146719481750478	0.00110335193761416	0.00155378449941	0.00185190976109158	0.00078437980830095	0.002297390761	0.000593301969003744	0.00100660299439846	0.00100982126886936	0.00106140941403487
0.00047687077958896	0.00037254399126128	0.0004953208357075	0.00062433816900479	0.00022861178803485	0.000593301969003744	0.001299386209	0.00047520985581144	0.000272862110406222	0.000338870814482629
0.00098188885956672	0.000802144011322368	0.00102529317381	0.0013825832274104	0.0005204185471224	0.00100660299439846	0.00047520985581144	0.001285079104	0.0006669093796005792	0.00070077383288712
0.00106224738710244	0.00098190595581408	0.0009320333733975	0.0010427022267247	0.0013229517991865	0.0005500131281865	0.000272862110406222	0.00066698796005792	0.001351444644	0.000882858808441304
0.0016583132963079	0.00117758246779734	0.00176839816159583	0.002545101601	0.0006555191965245	0.00174365408530377	0.00058871631196069	0.00123076274154484	0.00104270222672477	0.00103344261763808
0.00164119287675744	0.001221126503	0.00180780172337188	0.00195198527459366	0.00284462225	0.00082857213382025	0.00062433816990479	0.00113825832274104	0.00113229517991865	0.0010747274762908
0.0005616906078672	0.0005577811195824	0.000609895481625	0.0006555191965245	0.0020657025	0.00078437980830095	0.00022861178803485	0.0005204185471224	0.0005500131281865	0.0004889252847636
0.00146719481750478	0.00110335193761416	0.00155378449941	0.00185190976109158	0.00078437980830095	0.002297390761	0.000593301969003744	0.00100660299439846	0.00100982126886936	0.00106140941403487
0.00047687077958896	0.00037254399126128	0.0004953208357075	0.00062433816900479	0.00022861178803485	0.000593301969003744	0.001299386209	0.00047520985581144	0.000272862110406222	0.000338870814482629
0.00098188885956672	0.000802144011322368	0.00102529317381	0.0013825832274104	0.0005204185471224	0.00100660299439846	0.00047520985581144	0.001285079104	0.0006669093796005792	0.00070077383288712
0.00106224738710244	0.00098190595581408	0.0009320333733975	0.0010427022267247	0.0013229517991865	0.0005500131281865	0.000272862110406222	0.00066698796005792	0.001351444644	0.000882858808441304
0.0016583132963079	0.00117758246779734	0.00176839816159583	0.002545101601	0.0006555191965245	0.00174365408530377	0.00058871631196069	0.00123076274154484	0.00104270222672477	0.00103344261763808
0.00164119287675744	0.001221126503	0.00180780172337188	0.00195198527459366	0.00284462225	0.00082857213382025	0.00062433816990479	0.00113825832274104	0.00113229517991865	0.0010747274762908
0.0005616906078672	0.0005577811195824	0.000609895481625	0.0006555191965245	0.0020657025	0.00078437980830095	0.00022861178803485	0.0005204185471224	0.0005500131281865	0.0004889252847636
0.00146719481750478	0.00110335193761416	0.00155378449941	0.00185190976109158	0.00078437980830095	0.002297390761	0.000593301969003744	0.00100660299439846	0.00100982126886936	0.00106140941403487
0.00047687077958896	0.00037254399126128	0.0004953208357075	0.00062433816900479	0.00022861178803485	0.000593301969003744	0.001299386209	0.00047520985581144	0.000272862110406222	0.000338870814482629
0.00098188885956672	0.000802144011322368	0.00102529317381	0.0013825832274104	0.0005204185471224	0.00100660299439846	0.00047520985581144	0.001285079104	0.0006669093796005792	0.00070077383288712
0.00106224738710244	0.00098190595581408	0.0009320333733975	0.0010427022267247	0.0013229517991865	0.0005500131281865	0.000272862110406222	0.00066698796005792	0.001351444644	0.000882858808441304
0.0016583132963079	0.00117758246779734	0.00176839816159583	0.002545101601	0.0006555191965245	0.00174365408530377	0.00058871631196069	0.00123076274154484	0.00104270222672477	0.00103344261763808
0.00164119287675744	0.001221126503	0.00180780172337188	0.00195198527459366	0.00284462225	0.00082857213382025	0.00062433816990479	0.00113825832274104	0.00113229517991865	0.0010747274762908
0.0005616906078672	0.0005577811195824	0.000609895481625	0.0006555191965245	0.0020657025	0.00078437980830095	0.00022861178803485	0.0005204185471224	0.0005500131281865	0.0004889252847636
0.00146719481750478	0.00110335193761416	0.00155378449941	0.00185190976109158	0.00078437980830095	0.002297390761	0.000593301969003744	0.00100660299439846	0.00100982126886936	0.00106140941403487
0.00047687077958896	0.00037254399126128	0.0004953208357075	0.00062433816900479	0.00022861178803485	0.000593301969003744	0.001299386209	0.00047520985581144	0.000272862110406222	0.000338870814482629
0.00098188885956672	0.000802144011322368	0.00102529317381	0.0013825832274104	0.0005204185471224	0.00100660299439846	0.00047520985581144	0.001285079104	0.0006669093796005792	0.00070077383288712
0.00106224738710244	0.00098190595581408	0.0009320333733975	0.0010427022267247	0.0013229517991865	0.0005500131281865	0.000272862110406222	0.00066698796005792	0.001351444644	0.000882858808441304
0.0016583132963079	0.00117758246779734	0.00176839816159583	0.002545101601	0.0006555191965245	0.00174365408530377	0.00058871631196069	0.00123076274154484	0.00104270222672477	0.00103344261763808
0.00164119287675744	0.001221126503	0.00180780172337188	0.00195198527459366	0.00284462225	0.00082857213382025	0.00062433816990479	0.00113825832274104	0.00113229517991865	0.0010747274762908
0.0005616906078672	0.0005577811195824	0.000609895481625	0.0006555191965245	0.0020657025	0.00078437980830095	0.00022861178803485	0.0005204185471224	0.0005500131281865	0.0004889252847636
0.00146719481750478	0.00110335193761416	0.00155378449941	0.00185190976109158	0.00078437980830095	0.002297390761	0.000593301969003744	0.00100660299439846	0.00100982126886936	0.00106140941403487
0.00047687077958896	0.00037254399126128	0.0004953208357075	0.00062433816900479	0.00022861178803485	0.000593301969003744	0.001299386209	0.00047520985581144	0.000272862110406222	0.000338870814482629
0.00098188885956672	0.000802144011322368	0.00102529317381	0.0013825832274104	0.00052041					

4.6 Conclusion

Dans ce chapitre, on a résolu le modèle de Markowitz, la résolution est fait par la méthode FA programmé sur MATLAB, l'objectif de trouvé Meilleure solution avec des coefficients λ différents à chaque génération.

Conclusion général

Dans ce travail, nous avons étudié le problème d'optimisation de portefeuille, coté théorie et pratique, sous un aspect multi-objectif quadratique. Cela nous a conduits à faire une étude de l'optimisation multi-objectif, et de même l'optimisation quadratique pour fournir aux décideurs une bonne politique de gestion prudentielle, permet d'assurer le rendement optimal du portefeuille et de réduire les risques financiers.

La modélisation du problème posé nous a amené à résoudre, un modèle multi-objectif convexe pour le choix d'un portefeuille optimal.

Dans le but de réaliser ce travail, on a suivi le processus suivant :

Dans le premier chapitre nous avons commencé par quelques définitions de base et concepts nécessaires de problème d'optimisation combinatoire et de la gestion de portefeuille, on a cité la théorie moderne de Markowitz

puis, dans le deuxième chapitre, on a parler sur les metaheuristique

dans le troisième chapitre une présentation de notions de base de l'optimisation multi-objectif, et de quelques méthodes de résolution exactes et approchées.

Enfin dans le dernier chapitre, on a présenté une méthode de résolution approché Fire-Fly pour notre problème , on l'a implémenté sous le langage de programmation MATLAB. D'après les résultats obtenus, on déduit que notre méthode de résolution est efficace et fournit les meilleurs résultats et donne plus de choix aux décideurs.

Comme perspectives :

Améliorer la complexité de l'algorithme FireFly

Développer d'autres mécanismes de diversification plus performants, et l'approfondissement de notre étude en cherchant d'autres pistes plus poussées, en vue de déterminer le seuil optimum de diversification des portefeuilles.

Bibliographie

- [1] Saadi Leila. Optimisation MultiObjectifs par Programmation Génétique, mémoire de Magister, Université de Batna, 08 /07 /2007.
- [2] Abdesslem LAYEB, Utilisation des Approches d'optimisation Combinatoire pour la Vérification des Applications Temps Réel, thèse de doctorat université de Constantine 2, 27/05/2010
- [3] Amira Gherboudj, Méthodes de résolution de problèmes difficiles académique, Thèse de doctorat Université de Constantine2, 2013.
- [4] S.Arora and B.Barak. computational complexity : a modern approach. cambridge University Press 2009
- [5] Laurent Noé. Recherche de similarités dans les séquences d'ADN : modèles et algorithmes pour la conception de graines efficaces.2005.
- [6] Cours optimisation chapitre 4 problèmes np-complets
- [7] H. Markowitz. "Portfolio Selection". Dans : The Journal of Finance 7 Volume 7.1 (March. 1952), p. 77–91. doi : 10.1111/j.1540-6261.1952.tb01525.x.
- [8] A. Corhay et M. Mbangala. Fondements de gestion financière : manuel et applications. Boulevard Frère-Orban, Belgique : Editions de l'Université de Liège, 2007.
- [9] R. Bellman. Dynamic programming and stochastic control processes. Information and Control, 1(3) :228–239, 1958
- [10] Abdesslem LAYEB, Utilisation des Approches d'Optimisation Combinatoire pour la Vérification des Applications Temps Réel, thèse de doctorat université de Constantine 2, 27/05/2010
- [11] C.H. Papadimitriou, K. Steiglitz, Combinatorial optimization - algorithms and complexity. Prentice Hall, 1982.
- [12] MAHDI Samir Optimisation Multiobjectif Par Un Nouveau Schéma De Coopération Méta/Exacte, Mémoire de Magister, Université Mentouri de Constantine.
- [13] MAHDI Samir Optimisation Multiobjectif Par Un Nouveau Schéma De Coopération Méta/Exacte, Mémoire de Magister, Université Mentouri de Constantine.
- [14] Inès Alaya. Optimisation multi-objectif par colonies de fourmis : cas des problèmes de sac à dos. Ordinateur et société [cs.CY]. Université Claude Bernard - Lyon I, 2009. Français.

-
- [15] Les algorithmes gloutons, cour de formations.telecom-bretagne, page37.
- [16] P.Slavik. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms*, 25(2) : 237-254, 1997
- [17] P.E.Hart, N.J.Nilsson, and B. Raphael, A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions, on systems science and cybernetics* 4(2) : 100-107; 1968
- [18] W.Zeng and R.L Church. Finding shortest paths on real road network : the case for a. *International journal of geographical information science* 23(4) :531-543; 2009
- [19] M. Naimi. Amélioration de la Performance des Algorithmes Evolutionnaires Multiobjectifs : Application au Problème de Sac à Dos Multidimensionnel Multiobjectif. Thèse de doctorat. Université Hassan 2 Casablanca, Maroc. 2008.
- [20] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi. Optimization by simulated annealing. *Journal of Science*. Vol. 220, N° 4598, pp. 671-680, 1983.
- [21] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E.Teller. Equation of state calculations by fast computing machines. *Journal of Chem. Physics*. Vol. 21, N° 66, pp. 1087-1092, 1953.
- [22] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*. Vol. 13, N° 5, pp 533-549, 1986.
- [23] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston.1997.
- [24] N. Mladenovic, P. Hansen. Variable neighbourhood arc routing problem. *Computers and Operations Research*. Vol. 34, pp. 1097-1100, 1997.
- [25] C. Darwin. *The origin of species by means of natural selection*. Mentor Reprint, New York, 1859.
- [26] I. Rechenberg. *Cybernetic Solution Path of an Experimental Problem*, Royal Aircraft Establishment Library Translation, 1965 et H.G. Beyer. *The theory of evolution strategies*. Natural Computing Series, ISBN : 9783540672975. Springer, 2001.
- [27] L.J. Fogel, A.J. Owens, M.J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley Sons, Inc., New York, 1966.
- [28] J.H. Holland. Genetic Algorithms and the optimal allocation of trials, *SIAM Journal of Computing*. Vol. 2, N° 2, pp. 88-105, 1973.et D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*, ISBN : 0201157675, Addison-Wesley, 1989.
- [29] .Koza. *Genetic Programming. On the Programming of Computers by Means of Natural Selection*. A Bradford Book. MIT Press, 1996.
- [30] J.H. Holland. Genetic Algorithms and the optimal allocation of trials, *SIAM Journal of Computing*. Vol. 2, N° 2, pp. 88-105, 1973.
- [31] C. Darwin. *The origin of species by means of natural selection*. Mentor Reprint, New York, 1859.
- [32] K. Ghali. *Méthodologie de conceptionsystème a base de plateformes reconfigurables et programmables*. Thèse de doctorat. Université Paris XI, France. 2005.
- [33] A. Layeb. *Utilisation des Approches d'Optimisation Combinatoire pour la Vérification des Applications Temps Réel*. Thèse de doctorat. Université de Constantine 2, Algérie. 2010.
-

-
- [34] K. A. De Jong. An analysis of the behavior of a class of genetic adaptive systems. Ph. D. Thesis, University of Michigan, 1975.
- [35] M. Naimi. Amélioration de la Performance des Algorithmes Evolutionnaires Multiobjectifs : Application au Problème de Sac à Dos Multidimensionnel Multiobjectif. Thèse de doctorat. Université Hassan 2 Casablanca, Maroc. 2008.
- [36] A. Layeb. Utilisation des Approches d'Optimisation Combinatoire pour la Vérification des Applications Temps Réel. Thèse de doctorat. Université de Constantine 2, Algérie. 2010.
- [37] Z. W. Geem, J. H. Kim and G. V. Loganathan. A New Heuristic Optimization Algorithm : Harmony Search. *Simulation*. Vol. 76, N° 2, pp. 60-68, 2001. et Z. W. Geem and J.Y Choi. Music composition Using Harmony Search Algorithm. M. Giacobini et al. (Eds.) : *Evo-Workshops 2007*, LNCS 4448, pp. 593- 600. Springer, 2007.
- [38] Yang X.S. : "Nature-Inspired Metaheuristic Algorithms". Luniver Press, UK. (2008)
- [39] Yang X.S. : "Nature-Inspired Metaheuristic Algorithms". Luniver Press, UK. (2008). et Firefly algorithms for multimodal optimization. In : *Stochastic Algorithms : Foundations and Applications, SAGA 2009*. *Lectur Notes in Computer Sciences*, vol. 5792, pp. 169-178 (2009).
- [40] Jervase, J.A.; Bourdoucen, H.; Al-Lawati, A. Solar cell parameter extraction using genetic algorithms. *Meas. Sci. Technol.* 2001, 12, 1922.
- [41] Yang, X.-S. *Nature-Inspired Metaheuristic Algorithms*; Luniver Press : Frome, UK, 2010. et *engineering Optimization : An Introduction with Metaheuristic Applications*; Wiley : Hoboken, NJ, USA, 2010.
- [42] Yang, X.-S. *Nature-Inspired Metaheuristic Algorithms*; Luniver Press : Frome, UK, 2010.
- [43] Yang, X.-S. *Engineering Optimization : An Introduction with Metaheuristic Applications*; Wiley : Hoboken, NJ, USA, 2010.
- [44] Benlahrach. N. Optimisation multi-objectifs pour l'alignement Multiple de sequence. Mémoire de Magistère
- [45] Jozefowicz, N. Optimisation combinatoire multi-objectif : des méthodes aux problèmes, de la Terre à (presque) la Lune, Thèse d'habilitation. Institut National Polytechnique de Toulouse, 2013
- [46] Merdjaoui, M. optimisation multi-objectif par l'algorithme génétiques et approches Pareto des paramètres d'usinage sans contraintes des limitations de production. Mémoire de Magistère, Université M'hemed Bougara de Boumerdes, 2006
- [47] Knowles, J. Corne, D and Deb, K. *Multiobjective Problem Solving from Nature*. Springer, Berlin, 2008.
- [48] Sébastien. A. (Septembre 2011). Documents, Graphes et Optimisation Multi-Objectifs. Rapport de recherche. L'Université de Rouen.
- [49] Miettinen, K et Mäkelä, M. Proper Pareto Optimality In Nonconvex Problems Characterization With Tangent And Normal Cones. Technical report, UNIVERSITY OF JYVÄSKYLÄ, 1998.
- [50] Golberg D. E. *Genetic algorithms in Search, Optimization and Machine learning*. AddisonWesley Publishi
-

- [51] ZERDANI. O. (2013). L'optimisation non Linéaire Multi-objectif. Thèse de Doctorat. Université MOULOUD Mammeri, Tizi-ouzou
- [52] Mahdi, S. Optimisation Multi-objectif Par Un Nouveau Schéma De Coopération Méta/Exacte. Mémoire de Magistère, Université de Constantine, 2007.