République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Djilali BOUNAÂMA Khemis Miliana

Faculté des Sciences et de la Technologie

Département de Mathématiques et d'Informatique

Mémoire Présenté

Pour l'obtention de diplôme de

Master en« Informatique»

Spécialité : «Ingénierie du logiciel»

Titre :

Approche évolutionnaire multi-objectif

Réalisé par :
-Khalal Ilhem
-Ben Halima Souhila.

Soutenu publiquement le :21/09/2019

devant le jury composé de:

Mr.Hadj Sadok……..Président

Mme.Maghatria……..……Encadreur

Mr Sakri…….Examinateur1

Année Universitaire 2018/2019

Dédicaces

Je dédie ce modeste travail aux deux êtres les plus chers à mon cœur auxquels je dois mon existence «♥Mon Père et ♥Ma Mère »

Vous qui étaient toujours à mes côtés pour me soutenir et m'encourager à me battre sans jamais m'arrêter à mi-chemin, que Dieu vous protège.

♥À mes chères filles : NourElhoda ,Hadile ,Nihale et Maria.

♥À mes chères frères et ♥À mes chères sœurs

♥À tous mes amis de la promotion 2ème année master informatique 2018/2019 et à tous ceux qui m'ont aidé pour la réalisation de ce travail :

À ceux-là, et à tous ceux que j'aurais oublié de citer, j'exprime mon infaillible reconnaissance et ma sincère gratitude.

Ilhem

Dédicaces

Je dédie modestie travail à ma petite famille mes chers parents, mon cher père et ma mère. C'est grâce à eux qui par leur aide et soutien, j'ai attient ce que je suis maintenant.

A mes adorables sœur « Lamya, Asma, et Malek »,

sans oublier mon unique frère « Youcef ».

A mes enseignants à leur tête mon promoteur « Mme.Maghatria »**,**

A tous mes proches, ainsi que tous ceux qui me connaissent, de près ou de loin.

A tous mes collègues qui m'ont accompagné durant ma carrière universitaire.

Souhila

Remerciements

Nous remercions Allah le tout puissant de nous avoir donné la volonté et le courage pour mener à bien ce travail.

Nous tenons à remercier vivement notre promoteur

«Mme Maghatria**»,** pour avoir accepté de diriger ce travail et pour ses précieux conseils et ses encouragements.

Nos vifs remerciements s'adressent à tous les membres de jury qui nous ont fait l'honneur d'examiner ce travail.

Nous remercions également tous nos enseignants pour toutes les connaissances qu'ils nous ont inculquées ;

Enfin, nous remercions tous nos amis qui nous ont aidé, encouragé et toute personne ayant contribué à l'élaboration de ce travail, par un conseil, ou même par un souri

INTRODUCTION GENERALE **10**

CHAPITRE I : NOTION DE BASE EN OPTIMISATION **12**

I.1 INTRODUCTION : 14
I.2.DEFINITIONS D'OPTIMISATION (DICTIONNAIRE). 14
I.3. DEFINITION DES PROBLEMES D'OPTIMISATION 15
I.4.DEFINITION DE L'OPTIMUM GLOBALE ET LOCALE 15
I.5. Méthodes évolutionnistes *15*
I.6 PRINCIPES DES ALGORITHMES EVOLUTIONNAIRES 16
I.8 ALGORITHME ..... - 17 -
I.9 HEURISTIQUE ..... - 17 -

CHAPITRE II : ALGORITHMES GENETIQUES ..... **- 18 -**

II.1.INTRODUCTION ..... - 20 -
II.3. FONCTIONNEMENT DES AGS : ..... - 20 -
II.4.REPRESENTATION CHROMOSOMIQUE(CODAGE) : ..... - 21 -
II.4.1.Codage binaire : ..... *- 22 -*
II.4.2.Codage réel : *23*
II.4.3.Codage gray : *23*
II.5.EVALUATION : 24
II.6.SELECTION : 24
II.6.1. Méthode de la loterie biaisée (roullet) : *24*
II.6.2.Méthode de sélection par rang : *25*
II.7.LES OPERATEURS DE GENETIQUES : ..... - 26 -
II.7.1.Croisement : ..... *- 26 -*
II.7.2.Mutation : ..... *- 27 -*
II.8. APPLICATION SUR QUELQUES FONCTIONS DE TEST [22]: ..... - 27 -
II. 9.ALGORITHMES GENETIQUES MULTI-OBJECTIFS ..... - 30 -
II.9.1.Problématique ..... *- 30 -*
II.9.2. Résoudre un problème Multi- objectif ..... *- 30 -*
II.9.3. Familles des méthodes d'optimisation multi-objectif ..... *- 31 -*
II.9.3.1 Les méthodes a priori (décideur → recherche) ..... *- 32 -*
II.9.3.2 Les méthodes a posteriori (recherche → décideur) ..... *- 32 -*
II.9.3.3 Les méthodes progressives ou interactives (décideur ↔ recherche) ..... *- 32 -*
II.9.4.Les méthodes de résolution des problèmes d'optimisation multi-objectif ..... *- 33 -*
II.9.4.1 Les méthodes agrégées ..... *- 33 -*
II.9.4.2 Les méthodes non agrégées et non Pareto ..... *- 34 -*
II.9.4.3 Les méthodes fondées sur Pareto ..... *- 35 -*
II.9.4.4 Techniques non élitistes ..... *- 38 -*
II.9.4.5 Techniques élitistes ..... *- 40 -*
II.10 Conclusion ..... *- 43 -*

CHAPITRE III : Contribution et Implémentation ..... ***-41-***

III.1.Introduction : ..... *-41-*
III.2.Echantillonnage : ..... *-42-*
III.3. Méthodes d'échantillonnage : ..... *-42-*
III.3.1. Méthodes probabilistes (Aléatoires) : ..... *-42-*
III.3.2. Echantillonnage aléatoire simple : ..... *-42-*
III.3.3. Echantillonnage aléatoire stratifie : ..... *-42-*
III.3.4. Echantillonnage aléatoire par grappe : ..... *-43-*
III.3.5. Echantillonnage aléatoire systématique : ..... *-43-*
III.3.6. Méthodes non probabilistes (Raisonnées ou empirique) : ..... *-43-*

III.4.Outils utilisés :……*-44-..*
III.4.1. Outils matériels :…… *-44-*
III.4.2. Outils logiciels :…… *-44-*
III.5. Organigramme ……*-45-*
III .6 Algorithme ……*-45-*
III.7. Fonctions objectifs (fitness) ……*46--*
III.8. Méthode de résolution des problèmes d'optimisation multi-objectif……*-46-*
III 9. Implémentation ……*-48-*
III.9.1 Programme principale……*-48-*
III.10 Coclusion……*-54-*

CONCLUSION GENERALE*……...*-56-

REFERENCES BIBLIOGRAPHIQUES : ..... - 62 -

Liste des tableaux

Tableau II.1- Codage Binaire/Gray…….

Tableau II.2 Paramètres de l'AG mono-objectif……

Tableau II.3 Comparaison des résultats (McCormick)……

Tableau III.1 Population initial…….….53

Tableau III.2 Résultats final d'évaluation d'échantillon Population initial…….53

Liste des figures

Figure I.1 Fonctionnement général d'un algorithme évolutionnaire [18].
Figure II.1 Fonctionnement de l'algorithme génétique
Figure II.2. Représentation Chromosomique
Figure II.3 Codage Binaire de trois variables
Figure II.4Codage Réel
Figure II.5 Méthode de sélection de la loterie biaisée
Figure II.6 Croisement en un seul point
Figure II.7.Croisement en deux points
Figure II.8.Mutation d'un chromosome
Figure II.9 Fonction de McCormick
Figure II.10 (McCormick) Evolution de l'optimum en fonction des générations……27
Figure II.11 Modes de résolution d'un problème Multi objectif [17].
Figure II.12 Schéma de fonctionnement de VEGA [16].
Figure II.13 : Exemple de dominance [16].
Figure II.14 Exemples de frontière de Pareto [16].
Figure II.15 : La représentation de la surface de compromis [20].
Figure II.16 : Schéma de fonctionnement de PESA.
Figure II.17 : Mesure du squeeze_factor de PESA……..……..40
Figure III.1 : logo Matlab (version R2013a)……..…..……...45
Figure III.2 organigramme ……...46
Figure1 : R1= 11.0182……..……...……54
Figure2 : R2= 2.8241 ……..……...54
Figure 3 : R3= 0.8241……54
Figure 4 : R4= 0.8241……..…….……...54
Figure 5 : R5= 0.8241……..…….. 54

الملخص

في معظم المشكلات الحقيقية ، سواء كانت تقنية أو اقتصادية أو غير ذلك ، فإن مهارة اتخاذ القرار ضرورية.
يجب أن يكون لدى صانع القرار معرفة جيدة جدًا بالمشكلة حتى يمكن أن تكون الخيارات هي الأمثل من الناحية الموضوعية. ومع ذلك ، فإن مشاكل العالم الحقيقي لها العديد من الأهداف التي يجب تحقيقها ، لهذا يجب البحث عن الحلول التي تحقق هذه الأهداف في وقت واحد .
وفي هذا الصدد تم استخدام العديد من الطرق لحلها ، و تعد خوارزمياتها التطورية وخوارزمياتها الوراثية الأكثر دقة . ويرجع ذلك إلى قدرتهم على إيجاد الحلول المناسبة .
يتم تصنيف معظم الخوارزميات التطورية وفقًا لتدخل صانع القرار في ثلاث فئات: البدائية ، الخلفية والتفاعلية. توفر الخوارزمية الحلول المثلى لصانع القرار الذي سيختار الحلول المناسبة. في جميع الفئات ، إذا كان صانع القرار غير راض عن الحلول المقدمة ، فيجب إعادة تنفيذ الخوارزمية للإجاد حلول بديلة .
من أجل مواجهة هذه المشكلة ، يجب أن تكون الخوارزمية المستخدمة قادرة على دمج الحلول التي تم الحصول عليها خلال مرحلة التحسين من جل ذلك ، يتم استخدام خوارزمية جينية متعددة الأهداف. بهدف تقدير أفضل الحلول التي سيتم استخدامها لإنشاء حلول جديدة مع نفس خصائص الأمثلية . في هذه الرسالة ، أجريت دراسة تطورية متعددة الأهداف. وكذا أخذ عينات بهدف قليل وقت التنفيذ في مرحلة إيجاد الحلول المثلى.

- **كلمات البحث**: التحسين متعدد الأهداف ، خوارزمية التطورية ، خوارزمية الوراثية متعددة الاهداف ، أخد عينات

Résumé

Dans la plupart des problèmes réels que ce soit technologiques, économiques ou autres, une compétence du point de vue choix de décisions à faire est indispensable.
Un décideur doit avoir une très bonne connaissance autour du problème traité pour que les choix puissent être optimaux les objectifs à atteindre. Néanmoins, les problèmes du monde réel comportent plusieurs objectifs à atteindre, la recherche des solutions qui optimisent les objectifs simultanément rentre dans le domaine d'optimisation multi-objectif.
Plusieurs méthodes ont été utilisées pour résoudre ce problème, dont les algorithmes évolutionnaires et plus précis des algorithmes génétiques sont les mieux adaptés. Ceci est du à leurs capacités de trouver les bonnes approximations des solutions.
La plupart des algorithmes évolutionnaires sont classés selon la manière d'intervention du décideur en trois catégories : a priori, a posteriori et interactif. L'algorithme fournit des solutions optimales au décideur qui va choisir celles qu'il le convient. Dans toutes les catégories, si le décideur n'est pas satisfait par les solutions fournies, l'algorithme doit être ré-exécuté .
An de faire face à ce problème, l'algorithme utilisé doit être capable de modéliser les solutions obtenues lors de la phase d'optimisation. Un tel modèle sera exploité dans le cas où de nouvelles solutions sont requises.

Pour cela, un algorithme génétique multi-objectif est utilisé. Il vise à estimer de meilleures solutions qui seront utilisée pour générer des nouvelles solutions avec les mêmes caractéristiques d'optimalité. Dans cette thèse, une approche évolutionnaire multi-objectif aura proposer , afin de minimiser le temps d'exécution dans la phase de la recherche des solutions optimal .

- **Mots-clé** : Optimisation, métaheuristique, algorithme évolutionnaire, algorithme génétique multi-objectif ,échantillonnage.

Abstract

In most real problems, be they technological, economic or otherwise, a decision-making skill is essential.
A decision maker must have a very good knowledge of the problem so that the choices can be optimal objectives to achieve. Nevertheless, the problems of the real world have several objectives to achieve, the search for solutions that optimize the objectives simultaneously enters the field of multi-objective optimization.
Several methods have been used to solve this problem, whose evolutionary algorithms and more accurate genetic algorithms are best suited. This is due to their ability to find the right approximations of the solutions.
Most evolutionary algorithms are classified according to the decision maker's intervention in three categories: a priori, a posteriori and interactive. The algorithm provides optimal solutions to the decision maker who will choose the right ones. In all categories, if the decision maker is not satisfied by the solutions provided, the algorithm must be re-executed.
In order to cope with this problem, the algorithm used must be able to model the solutions obtained during the optimization phase. Such a model will be exploited in case new solutions are required.
For this, a multi-objective genetic algorithm is used. It aims to estimate better solutions that will be used to generate new solutions with the same optimality characteristics. In this thesis, an evolutionary multi-objective approach will have to propose, in order to minimize the execution time in the phase of the search for optimal solutions.

Keywords: Optimization, metaheuristics, evolutionary algorithm, multi-objective genetic algorithm, samplin

Introduction générale

L'optimisation multi objectif (MOO) est une branche de l'optimisation combinatoire dont la particularité est de chercher à optimiser simultanément plusieurs objectifs d'un même problème (contre un seul objectif pour l'optimisation combinatoire classique). Elle se distingue de l'optimisation multidisciplinaire par le fait que les objectifs à optimiser portent ici sur un seul problème.

En général, les algorithmes multi objectif (MOO) donnent des solutions Pareto-optimales, c'est à dire, il n'y a pas de solution préférée à ces solutions sur l'ensemble de tous les objectifs. Cependant, la propriété de Pareto-optimalité suppose que les objectifs ont une importance égale. Dans les faits, les décideur sont souvent capables de différencier les objectifs à travers des préférences.

Dans le domaine de l'optimisation combinatoire, un grand nombre d'heuristiques, qui produisent des solutions proches de l'optimum, ont été développées, mais la plupart d'entre elles ont été conçues spécifiquement pour un problème donné.

L'arrivée d'une nouvelle classe de méthodes, nommées méta-heuristiques, marque une réconciliation des deux domaines.

En effet, celles-ci s'appliquent à toutes sortes de problèmes combinatoires et elles peuvent également s'adapter aux problèmes continus. Ces méthodes, qui comprennent notamment la méthode du recuit simulé, les algorithmes génétiques, la méthode de recherche tabou, les algorithmes de colonies de fourmis…etc.

La principale difficulté que l'on rencontre en optimisation mono-objectif vient du fait que modéliser un problème sous la forme d'une équation unique peut être une tâche difficile. Avoir comme but de se ramener à une seule fonction objectif peut aussi biaiser la modélisation.

L'optimisation multi objectif autorise ces degrés de liberté qui manquaient en optimisation mono-objectif. Cette souplesse n'est pas sans conséquences sur la démarche à suivre pour chercher un optimum à notre problème, modélisé. La recherche ne nous donnera plus une solution unique mais un ensemble de solutions.

Ces problèmes d'optimisation appartiennent à la classe des problèmes NP-difficile :classe où il n'existe pas d'algorithme qui fournit la solution optimale en temps polynomial en fonction de la taille du problème.

Dans ce cas en proposer une solution pour minimiser le temps d'exécution par l'échantillonnage de la population initiale, l'utilisation des méthodes méta heuristiques (génériques) permettant d'approximer les meilleures solutions sur les plus grandes instances.

Le premier chapitre de notre travail et sur des notions générales en optimisation puits va parler sur les algorithmes génétiques multi objectif dans la deuxième chapitre et à la fin en présente notre travail en deux chapitres Contribution et implémentation avec une conclusion générale.

CHAPITRE I : Notion de base en optimisation

I.1 Introduction :

Dans ce chapitre introductif nous rappelons quelques notions de base qui seront utiles dans la suite de notre travail. Nous citons aussi quelques méthodes et approches d'optimisation locales et globales les plus connues.

I.1.Définitions d'optimisation (mathématiques).

L'optimisation est une branche des mathématiques cherchant à modéliser, à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à minimiser ou maximiser une fonction sur un ensemble [21].

I.2.Définitions d'optimisation (dictionnaire).

Raisonnement ou calcul permettant de trouver les valeurs d'un ou plusieurs paramètres qui correspondent au maximum d'une fonction [21].

I.3. Définition des problèmes d'optimisation

Considérons le problème général d'optimisation suivant :

$$\begin{cases} \min f(x) \\ x \in X \end{cases}$$

où f est la fonction objectif et X la région faisable.

Optimiser la fonction f consiste à déterminer les éléments $x \in X$ tels que f atteignesa valeur minimale (resp. sa valeur maximale) en x ; on parle alors de minimisation (resp. maximisation).

I.4.Définition de l'optimum globale et locale

On distingue deux types d'optimisation, locale et globale.

$-x^*$est un minimum local de f s'il existe un voisinage V (x) de x tel que

$$f(x^*) \leq f(x) \qquad \forall x \in V(x) \cap X$$

$-x^*$est un minimum global de f si

$$f(x^*) \leq f(x) \qquad \forall x \in X$$

I.5. Méthodes évolutionnistes

Les algorithmes évolutionnaires (AE) sont inspirés par des concepts issus de la théorie de l'évolution de Darwin. Ils sont des techniques d'optimisation itérative et stochastique, car ils utilisent itérativement des processus aléatoires. Ils font évoluer un ensemble de solutions (une population) à un problème donné en utilisant une fonction objectif afin de mesurer ses valeurs, dans l'optique de trouver les meilleurs résultats **[18].**

Le terme d'algorithme évolutionnaire est relatif à une classe de méthodes d'optimisation stochastiques qui simulent le processus de l'évolution naturelle. Les origines des algorithmes évolutionnaires remontent à la fin des années 50, et depuis 1970, plusieurs méthodes évolutionnaires ont été proposées, principalement concernant les algorithmes génétiques, la programmation génétique et les stratégies d'évolution.

Toutes ces approches opèrent sur un ensemble de solutions candidates. Utilisant d'importantes simplifications, cet ensemble est successivement modifié par deux principes de l'évolution : la sélection et la variation. La sélection représente la compétition pour les ressources parmi les êtres vivants. Certains sont meilleurs que d'autres et sont plus aptes à survivre et à transmettre leur matériel génétique. Dans les algorithmes évolutionnaires, la sélection naturelle est simulée par un processus de sélection stochastique.

Chaque solution a une chance de se reproduire un certain nombre de fois, dépendant

de leur qualité (fitness). Donc, la qualité est estimée en évaluant les individus et en leur assignant une valeur de fitness **[10] [15].**

I.6 Principes des algorithmes évolutionnaires

En général, un AE est caractérisé par trois faits :

1. A base de population (un ensemble de solutions candidates est maintenu).
2. celui-ci subi un processus de sélection (les mieux adaptés survivent).
3. Héritage de patrimoine génétique.
4. et manipulé par des opérateurs génétiques, le plus souvent le croisement et la mutation.
5. Amélioration globale.
6. Fonction objectif **[19].**

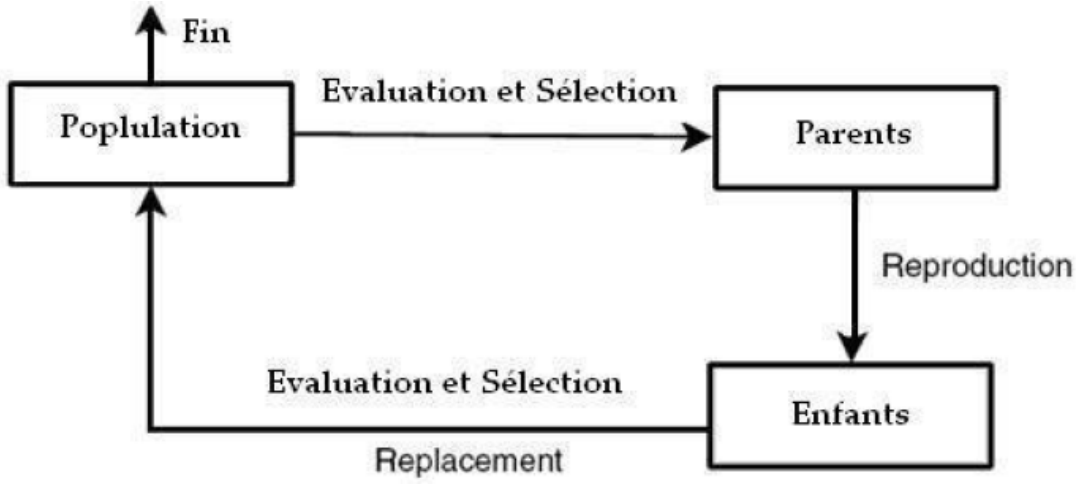

Figure I.1 : Fonctionnement général d'un algorithme évolutionnaire **[18].**

Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum par échantillonnage d'une fonction objectif. Les métaheuristiques les plus classiques sont celles fondées sur la notion de parcours. Dans cette optique, l'algorithme fait évoluer une seule solution sur l'espace
de recherche à chaque itération **[20].**

I.8 Algorithme

Un algorithme est un moyen de présenter la résolution par calcul d'un problème à une personne ou à une machine. Plus spécifiquement, il s'agit d'un énoncé, dans un langage bien défini, d'une suite d'opérations permettant de résoudre par calcul un problème **[20].**

I.9 Heuristique

Les heuristiques sont souvent, à la différence des algorithmes, tirées de l'expérience ou d'analogies, plutôt que d'une analyse scientifique trop complexe. Elles trouvent leur place dans les algorithmes qui nécessitent l'exploration d'un grand nombre de cas, car celles-ci permettent de réduire leur complexité moyenne en examinant d'abord les cas qui ont le plus de chances de donner la réponse **[20][21].**

CHAPITRE II : Algorithmes Génétiques.

II.1.Introduction

Les AGs (Holland, 1975 [10]) font partie des algorithmes méta-heuristiques, ils travaillent sur un espace de recherche sous forme de population de pseudo-solutions en y opérant d'une manière stochastique sans qu'il y ait de contraintes de continuité ni de différentiable de la fonction à optimiser, ce qui est d'ailleurs un des points qui différencie les AGs des autres méthodes classiques.

II.2. Définition

Les algorithmes génétiques (AGs) sont des méthodes d'optimisations numérique inspirée du processus naturel de l'évolution génétique (Darwin, 1859 [9]), c'est une méthode d'optimisation puissante, elle est appliquée à un large éventail de problèmes. Les AGssontreconnuspourleureffectivitédanslamajoritédescasetsonttoutletempsutilisés pour résoudre des problèmes pratiques.

II.3. Fonctionnement des AGs :

Semblables au processus naturel de l'évolution, les algorithmes génétiques comportent des phases à travers lesquelles une solution potentielle est estimée :

1- **Initialisation :** Un ensemble de N chromosomes est aléatoirement créé formant une population de solutions potentielles.
2- **Evaluation :** Décodage puis évaluation de chaque individus/chromosomes.
3- **Reproduction :** Après l'évaluation, une nouvelle population de N individus est créée à l'aide d'une méthode de sélection adéquate.
4- **Retour :** Tant que la condition d'arrêt n'est pas satisfaite, l'algorithme recommence à partir de la phase d'évaluation.

UnAGgénèreunepopulationenutilisantunereprésentationchromosomiquedesindividus puis à l'aide de la fonction objectif (fitness), il évalue chaque individu et sélectionne les mieux adaptés pour former une nouvelle population de solutions potentielles à travers les opérateurs génétiques (croisement, mutation) là ou se passe la reproduction afin de créer de nouveaux individus au sein de la nouvelle population.

À travers ces phases, le mécanisme de la sélection naturelle est simulé, afin d'assurer la survie, les AGs gardent à chaque fois les meilleurs individus en écartant les plus faibles.

Ce processus est illustré dans l'organigramme suivant :

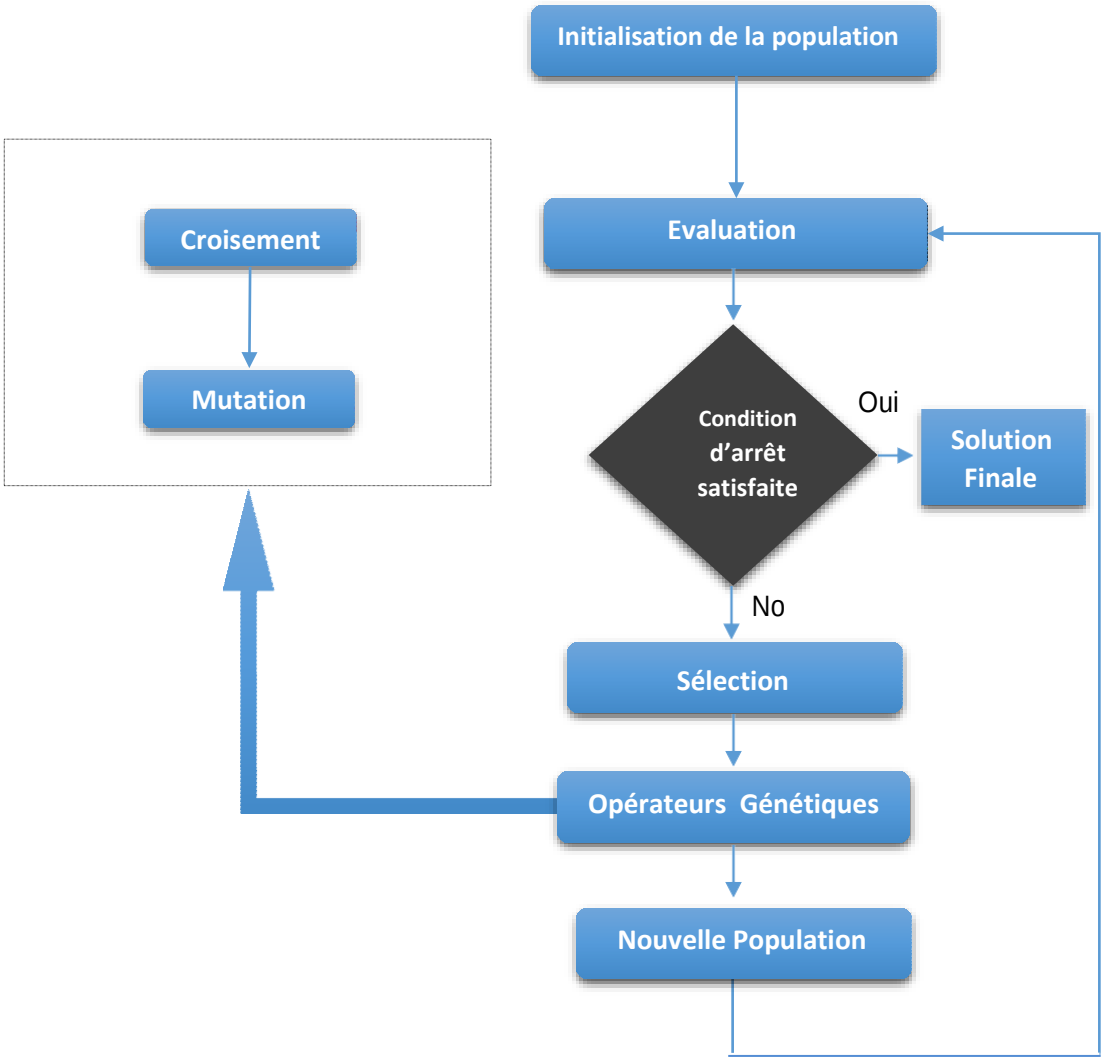

Figure II.1– Fonctionnement de l'algorithme génétique

II.4.Représentation chromosomique(Codage) :

Au début, chaque individu (chromosomique) de la population est codé soit en binaire/gray (Goldberg 1989 [11]), ou en réel (Goldberg 1990 [12]), représenté par un vecteur dont la longueur dépend de la précision requise, le codage donne à l'algorithme génétique une représentation génotype-phénotype.

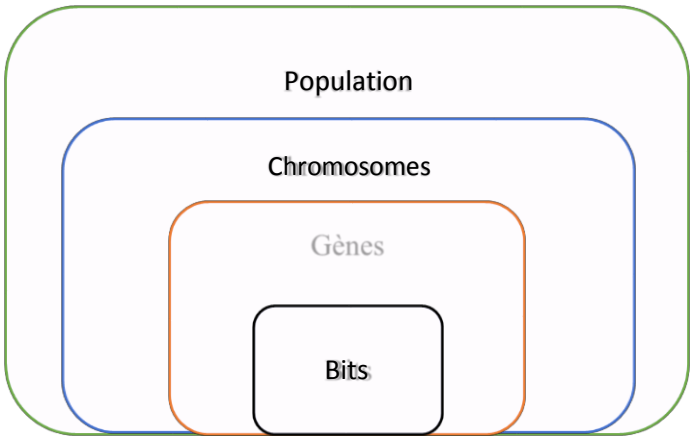

Figure II.2. Représentation Chromosomique

La rapidité de la convergence d'un AG peut être affectée par le choix initial de la population, car si des renseignements à priori sur le problème sont disponibles, l'orientation du choix de la population peut considérablement améliorer la rapidité du processus de résolution.

Le codage des paramètres d'un AG dépend du problème à optimiser, le codage binaire, gray et réel sont les plus utilisé dans le domaine.

II.4.1.Codage binaire :

Ce codage est caractérisé par sa simplicité car il ne comporte que deux caractères

(0 et 1), chaque individu est représenté par un chromosome et chaque caractéristique de l'individu est un gène.

Le chromosome est codé en une suite de bits appelée chaîne binaire, ceci est illustré à travers l'exemple suivant :

Trois variables (203, 13, 45) codées sur 8 bits formants une chaine binaire de taille 24 :

Chromosome

Gène1	Gène 2	Gène 3
11001011	00001101	00101101

Figure II.3 Codage Binaire de trois variables

II.4.2.Codage réel :

L'ensemble des variables est représenté par un vecteur$\bar{x} =< x1, x2, \ldots . xn >$où chaque xi est un nombre réel. Les opérateurs de croisement et de mutation seront alors soigneusementdéfinisafindeconserverchacunedesvariablesdanssondomaine, la plupart de ces opérateurs s'inspirent des opérateurs du codage binaire.

Chromosome

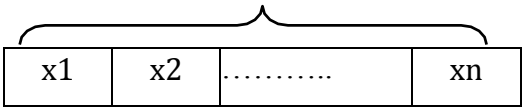

Figure II.4 Codage Réel

II.4.3.Codage gray :

Le codage gray a la propriété que deux points adjacents (n et n+1) dans l'espace étudié diffèrent seulement d'un seul bit, autrement dit, une incrémentation d'un pas dans la valeur du paramètre correspond à un changement d'un seul bit dans le code, ce qui est un avantage par rapport au codage binaire qui en terme de distance de Hamming ne code pas forcément deux éléments voisins dans l'espace d'exploration. Le tableau suivant illustre cette différence:

Valeur entire	Binaire	Gray
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111

Tableau II.1- Codage Binaire/Gray

II.5.Evaluation :

A l'aide d'une fonction objectif (Fitness), l'AG sera en mesure de classer les individus d'une population selon leurs pertinences, elle est une partie inséparable de l'AG car elle représente (sous forme mathématique) le problème à résoudre.

En général, l'algorithme génétique essai de minimiser la fonction de fitness en fonction d'un vecteur ou d'une valeur issus de la population, la solution est donc jugée selon son image par la fonction de fitness. Il existe des fonctions de test qui permettent de mettre en épreuve l'efficacité de l'algorithme génétique utilisé, certaines de ces fonction seront utilisées ultérieurement dans ce chapitre.

II.6.Sélection :

C'estdanscetteétapeoùsedécidequelssontlesindividusdelapopulationquivont être dupliqués dans la nouvelle population où ils deviendront des parents. Les individus ayants la meilleure fitness auront donc plus de chance d'être sélectionnés.

Il existe essentiellement quarte méthodes de sélection, leur utilisation dépend du problème étudié :

- La méthode de la « loterie biaisée » (roulette wheel) (Goldberg, 1989[11]).
- La méthode de rang (Baker, 1985[13]).
- La sélection par tournoi (Goldberg, Deb, 1991[14])
- La sélection universelle stochastique (Baker, 1987[14]).

II.6.1. Méthode de la loterie biaisée (roullet) :

C'est la méthode la plus utilisée, elle consiste à dupliquer des individus de la population dont les performances sont relativement bonnes.

Pour un chromosome dont la fitness est $f(ch_i)$, la probabilité(ch_i) avec la quelle il sera dupliqué dans la nouvelle population de taille N est :

$$p(ch_i) = \frac{f(ch_i)}{\sum_{j=1}^{N} f(ch_j)}$$

La figure suivante illustre le principe de ce type de sélection, en utilisant une population de quatre individus, l'individu 1 avec la probabilité (ch_i) = 0.37 a plus de chance d'être sélectionné que les autres.

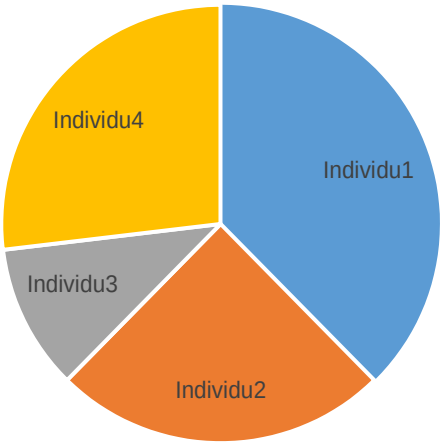

Figure II.5 Méthode de sélection de la loterie biaisée

II.6.2.Méthode de sélection par rang :

Le principe de cette méthode est de garder les meilleurs individus (selon leurs fitness) d'une population $P_t$pour ensuite les introduire dans la prochaine populationP_{t+1}. C'est une façon de protéger des solutions potentielles de la disparition lors de la phase de sélection. C'est une méthode performante, mais dans certains cas, le manque de diversité dans les solutions provoque la convergence prématurée ce qui représente l'inconvénient majeur de cette méthode de sélection.

II.6.3.Sélection par tour noir :

Cette méthode est une forme de duel entre deux individus, celui qui domine l'autre fonction de sa fitness est sélectionné pour être réintroduit dans la nouvelle population, le perdant est écarté. L'opération est répétée jusqu'à ce que la nouvelle population de nindividus soit pleine.

II.6.4.Méthode « stochastique du reste» :

Cette méthode repose principalement sur la fonction d'évaluation et sa moyenne numérique, chaque individu d'une population est dupliqué n_{a_i}fois avec n_{a_i}la valeur de la partie entière de $\left(\frac{f_i}{f}\right)$Chaque individu se fait associer une probabilité de sélection :

$$\mathbf{Ps(ai)} = \left(\frac{\mathbf{f_i}}{\mathbf{f}}\right) - \mathbf{E}\left(\mathbf{(fifmoy)}\right)\textbf{Avec E(x) La partie entière de x.}$$

II.7.Les Opérateurs de génétiques :

II.7.1.Croisement :

Pendantcetteopération,lapopulationestdiviséeendeuxsous-populations,en suite, l'opération de reproduction est appliquée sur chaque individu de la première sous- population avec un autre de la deuxième sous-population, chaque couple de parents forme deux nouveaux chromosomes (enfants) dont les caractéristiques sont issues des deux parents .Le taux de croisement est défini en général entre 0.5et 0.9,un taux excessivement élevé peut engendrer la perte de bonnes structures, mais aussi un taux faible provoque la stagnation puisque l'exploration est diminuée.

Deux méthodes classiques sont à mentionner, le croisement en un point et le croisement en deux points :

Chaque chromosome de longueur « l » est divisé en segments : deux segments pour le croisement en un seul point [16] et trois segments pour le croisementen deux points [23]). La position de chaque chromosome est tirée aléatoirement ensuite les gênes des individus parents sont échangés selon une probabilité de croisement $P_c$pour former deux nouveauchromosomes.Les schémas ci-dessous illustrent les deux méthodes :

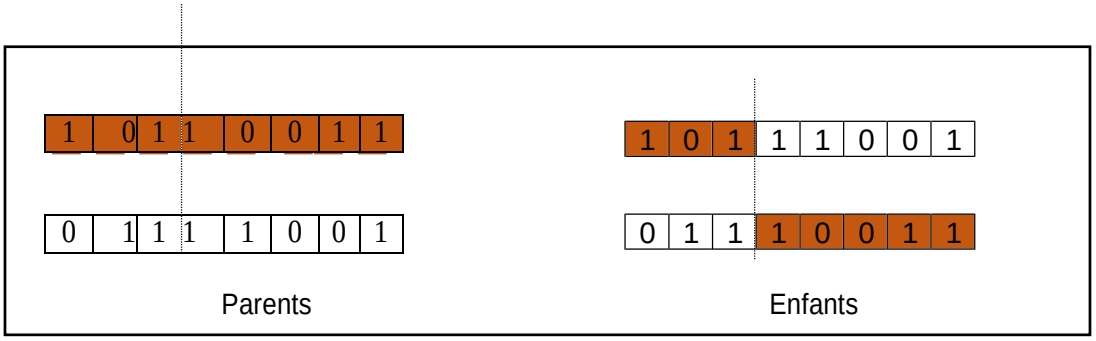

Figure II.6 Croisement en un seul point

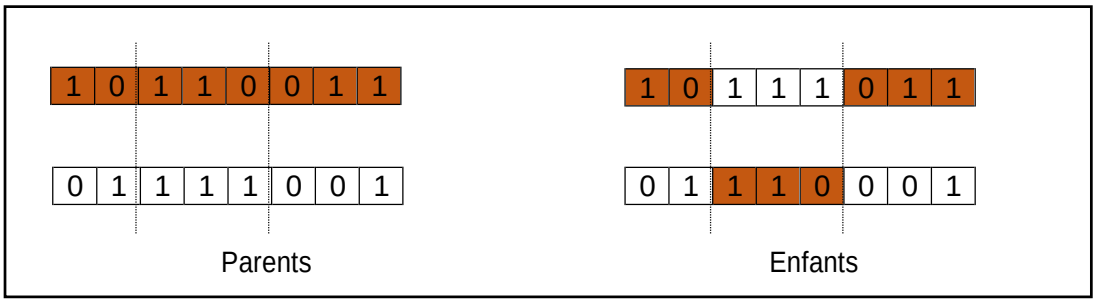

Figure II.7. Croisement en deux points

II.7.2.Mutation :

Cet opérateur tire au hasard un ou plusieurs gênes d'un individu, puis les modifie aléatoirement à condition que cet individu reste une solution possible au problème.

Chaque bit d'un individuel une probabilitéP_m des u bit une mutation, un réel r∈{0,1}est généré, si r <$P_m$le bit sera donc remplacé par complémentaire [16].

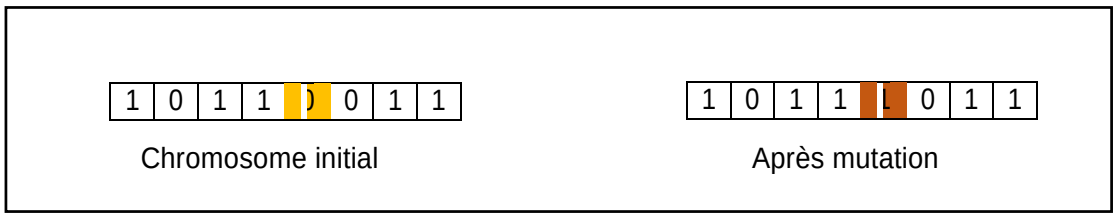

Figure II.8. Mutation d'un chromosome

L'opérateur de mutation rapporte de la diversité à la population, en évitant la convergence prématurée et donc l'AG peut atteindre la propriété d'ergodicité[2], c'est-à-dire que l'AG peut atteindre tous les points de l'espace de recherche et donc l'optimum global.

II.8. Application sur quelques fonctions de test [22]:

Dans cette partie, nous allons appliquer un algorithme génétique sur un problèmes d'optimisation la fonction de McCormick.

Codage	Binaire
Taille de population	80
Nombre de generations	80
Longueur du chromosome	8 bits
Sélection	Élitiste
Probabilité de mutation	0.02
Croisement	En un seul point avec p_c= 0.5

Tableau II.2 Paramètres de l'AG mono-objectif

- **Optimisation de la fonction de McCormick :**

Cette Fonction est définie par :

$$f(x1,x2)=\sin(x_1+x_2)+(x_1-x_2)^2-1.5x_1+2.5x_2+1$$

$$\text{avec}[x_1, x_2] \in [-1.5,4]$$

$$\text{Leminimumglobalest : } \min(x_1, x_2) = -1.9133 \quad \text{à} \quad [x_1, x_2] = [-0.54719, -1.54719]$$

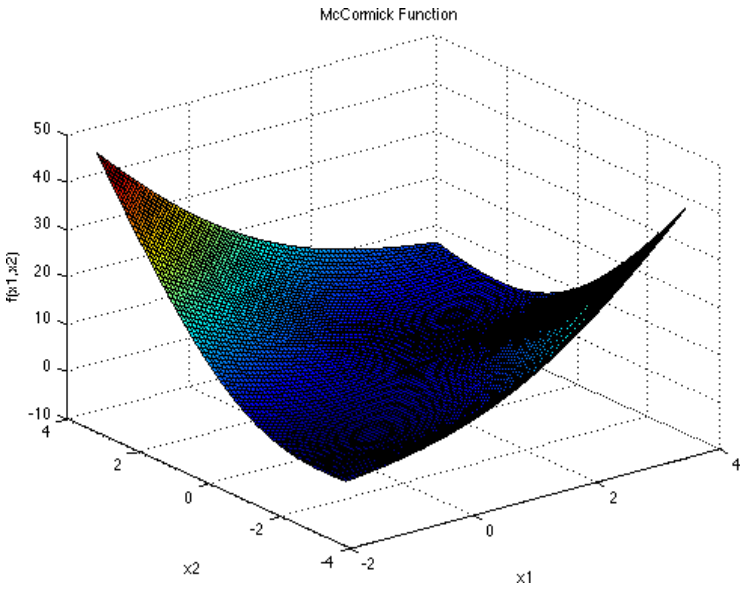

Figure II.9 Fonction de McCormick

Après l'exécution de l'algorithme génétique, ils ont obtenu les résultats suivants :

$$\min(x_1, x_2) = -1.9105$$

$$\text{Meilleuresolution} = [x_1, x_2] = [-0.52941, -1.5]$$

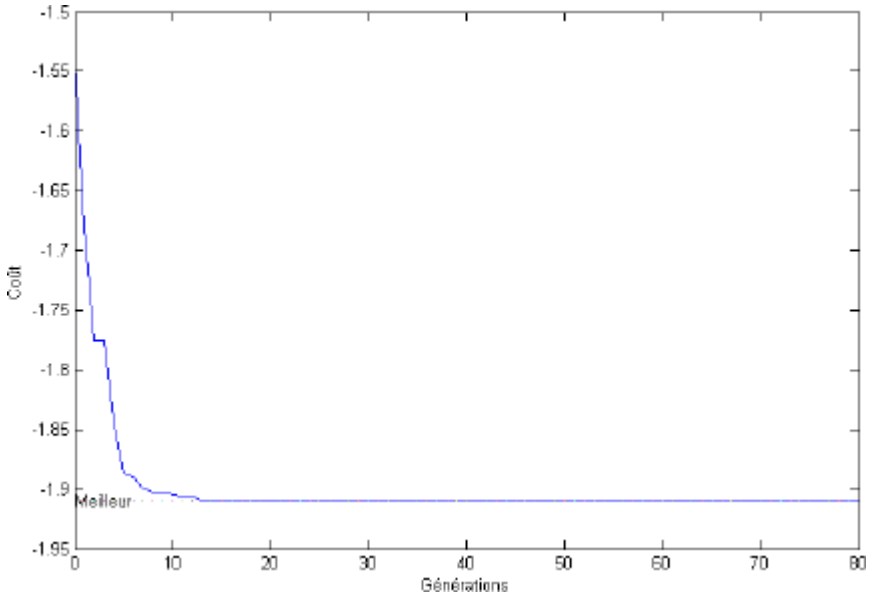

Figure II.10 (McCormick) Evolution de l'optimum en fonction des générations

D'après la figure II-10, la meilleure solution a été trouvée à la douzième génération, le résultat obtenu est très proche du minimum global :

Minimum global $[x_1,x_2]$	$[-0.54719, -1.54719]$
Résultatobtenu$[x_1,x_2]$	$[-0.52941, \quad -1.5]$

Tableau II.3 Comparaison des résultats (McCormick)

II. 9.Algorithmes génétiques multi-objectifs

L'optimisation multi objectif permet de modéliser des problèmes réels faisant intervenir de nombreux critères et contraintes. Dans ce contexte, la solution optimale recherchée n'est plus un simple point, mais un ensemble de bons compromis satisfaisant toutes les contraintes **[9].**

Un problème d'optimisation multi-objectif sous contraintes peut être défini comme suit :

$$\begin{cases} (\min|\max) f_i(\vec{x}) i = 1, \dots, k \\ C_I(\vec{x}) \geq 0 \quad I = 1, \dots, m \\ \vec{x} \in R^n \end{cases} \qquad (2.1)$$

Où « n » représente le nombre de variables, « x » un vecteur de décision, « k » le nombre d'objectifs (critères) et « m » le nombre de contraintes du problème.

Pour les problèmes d'optimisation multi objectif sous contraintes, il faut satisfaire un ensemble de contraintes tout en optimisant plusieurs fonctions objectives. En conséquence, même avec un petit nombre de variables et d'objectifs, le problème ne peut pas être traité simplement avec un algorithme classique de découpe. En effet, l'optimisation d'un problème multi-objectif est souvent plus difficile que l'optimisation des problèmes mono-objectifs car la solution optimale ne se réduit plus à un seul point, mais à un ensemble de points non dominés **[15].**

II.9.1.Problématique

La difficulté principale d'un problème multi objectif est qu'il n'existe pas de définition de la solution optimale. Le décideur peut simplement exprimer le fait qu'une solution est préférable à une autre mais il n'existe pas une solution meilleure que toutes les autres.

Dès lors résoudre un problème multi-objectif ne consiste pas à rechercher la solution optimale mais l'ensemble des solutions satisfaisantes pour lesquelles on ne pourra pas effectuer une opération de classement.

Les méthodes de résolution de problèmes multi objectifs sont donc des méthodes d'aide à la décision car le choix final sera laissé au décideur **[16].**

II.9.2. Résoudre un problème Multi- objectif

Deux types de comportement existent :

Le premier comportement : est de ramener un problème multi objectif à un problème simple objectif.

- **Le second comportement** : est de tenter d'apporter des réponses au problème au prenant en compte l'ensemble des critères.

La différence entre ces deux communautés s'exprime dans le schéma ci- dessous :

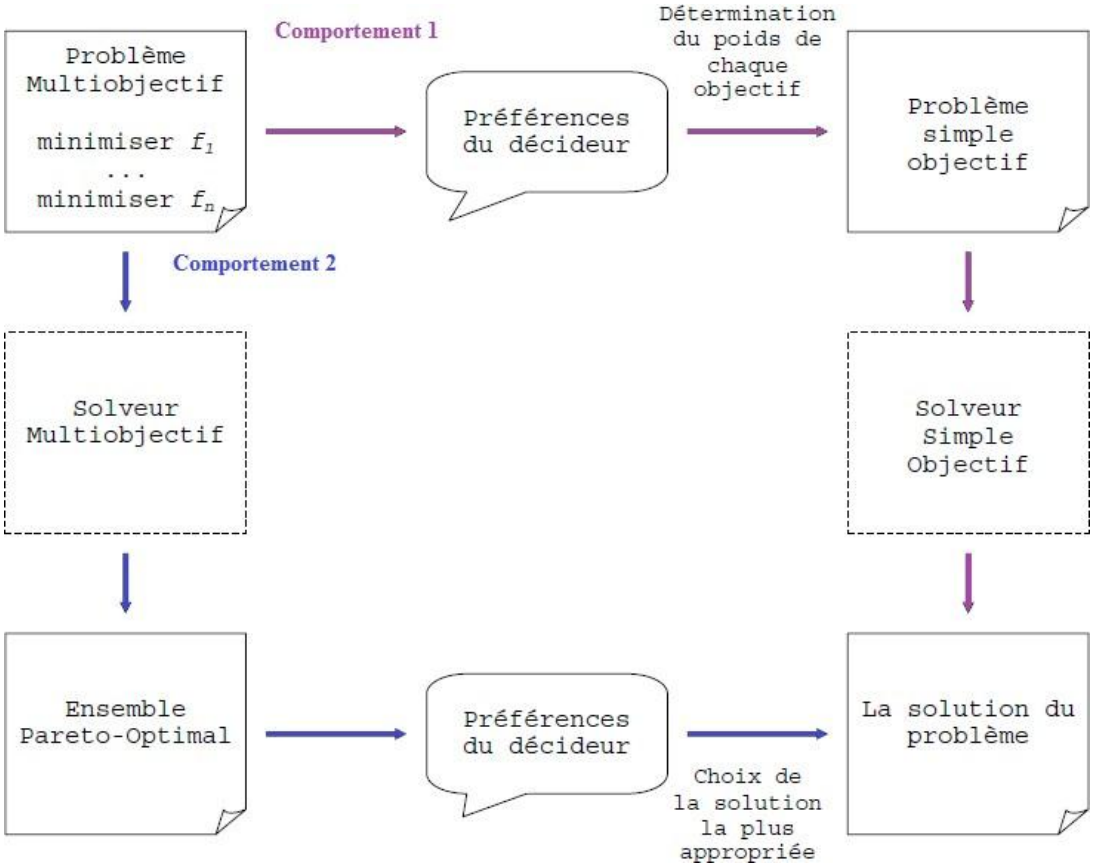

Figure II.11 Modes de résolution d'un problème Multi objectif **[17]**.

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 (Comportement 1).

Soit le décideur effectue son choix dans l'ensemble des solutions proposées par le solveur multi objectif (Comportement 2). La principale qualité d'un solveur multi objectif est donc de rendre les décisions plus faciles en proposant un sous-ensemble représentatif de F (l'ensemble des objectifs réalisables) **[16]**.

II.9.3. Familles des méthodes d'optimisation multi-objectif

On peut répartir les méthodes d'optimisation multi-objective en trois grandes familles. Ces familles sont définies en fonction de l'instant où l'on prend la décision d'effectuer ou non un compromis entre fonctions objectif. Nous avons les familles suivantes :

II.9.3.1 Les méthodes a priori (décideur → recherche)

Les solutions les plus intuitives pour résoudre des problèmes multi objectifs consistent souvent à combiner les différentes fonctions objectifs en une fonction d'utilité suivant les préférences du décideur. 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 simple objectif.

Cependant dans la plupart des cas, le décideur ne peut pas exprimer clairement sa fonction d'utilité, soit par manque d'expérience ou d'informations, soit parce que les différents objectifs sont non commensurables (ils sont exprimés dans des unités différentes) **[16]**.

II.9.3.2 Les méthodes a posteriori (recherche → décideur)

Le décideur prend sa décision d'après un ensemble de solutions calculées par un solveur. Dans ce cas la qualité de la décision dépend du choix de la méthode de résolution. Car celle-ci va devoir donner un ensemble de résultats le plus représentatif de l'espace des objectifs efficaces.**[16, 18]**.

Ici, il n'est plus nécessaire de modéliser les préférences du décideur. On se contente de produire un ensemble de solutions que l'on transmettra au décideur. Il y a donc un gain de temps non négligeable vis-à-vis de la phase de modélisation des préférences de la famille des méthodes a priori.

L'inconvénient qu'il faut souligner est que, maintenant, il faut générer un ensemble de solutions bien réparties. Cette tâche est non seulement difficile, mais, en plus, peut requérir un temps d'exécution important **[19].**

II.9.3.3 Les méthodes progressives ou interactives (décideur ↔ recherche)

Dans ces méthodes, les processus de décision et d'optimisation sont alternés. Par moment, 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 **[16]**.

Ces méthodes présentent l'inconvénient de monopoliser l'attention du décideurtoutaulongdel'optimisation.Cetinconvénientn'estpasmajeurlorsquel'on a affaire à des problèmes d'optimisation pour lesquels la durée d'une évaluation de la fonction objectif n'est pas importante.

Pour les autres problèmes, l'utilisation d'une telle méthode peut être délicate. Il est en effet difficile de monopoliser l'attention du décideur pendant une période qui peut s'étendre sur plusieurs heures en lui posant, de plus, des questions toutes les dix minutes, voire même toutes les

heures**[19].**

II.9.4.Les méthodes de résolution des problèmes d'optimisation multi-objectif

II.9.4.1 Les méthodes agrégées

Ces méthodes transforment un problème multi objectif en un problème simple objectif **[16]**.L'agrégation revient, généralement, à combiner différentes fonctions coûts de façon Linéaire. Elle s'applique à des problèmes dont les coefficients de pondération peuvent être définis et qui traduisent une évolution déterminée.

A. La moyenne pondérée :

Cette méthode consiste à additionner tous les objectifs en affectant à chacun d'eux un coefficient de poids. Ce coefficient représente l'importance relative que le décideur attribue à l'objectif. Cela modifie un problème multi objectif en un problème simple objectif de la forme :

$$\min \sum_{i=1}^{k} w_i f_i(x) \qquad \text{avec } w_i \geq 0$$

w_i représente le poids affecté au critère i et :

$$\sum_{i=1}^{k} w_i = 1$$

Critique :

Cette méthode est simple à mettre en œuvre et elle est d'une grande efficacité.

Mais les difficultés essentielles de cette approche sont :

1) Comment le décideur détermine-t-il les poids de chaque critère ?

2) Comment exprimer l'interaction entre les différents critères ?

B. Goal programming :

Cette méthode est également appelée « targetvector optimisation ».Le décideur fixe un but T_i à atteindre pour chaque objectif fi. Ces valeurs sont ensuite ajoutées au problème comme des contraintes supplémentaires. La nouvelle fonction objectif est modifiée de façon à minimiser la somme des écarts entre les résultats et les buts à atteindre :

$$\min \left| \sum_{i=1}^{k} f_i(x) - Ti \right| \quad \text{avec} \quad x \epsilon F$$

T_i représente la valeur à atteindre pour le $i^{ème}$ objectif.

Cette méthode a l'avantage de fournir un résultat même si un mauvais choix initial.

C. Le min-max :

Cette méthode est assez proche de la méthode « Goal programming ». Elle minimise le maximum de l'écart relatif entre un objectif et son but associé par le décideur :

$$\min \max \left(\frac{f_i(x) - T_i}{T_i} \right) \quad \text{avec} \quad i = 1 \dots k$$

T_i représente le but atteindre pour le $i^{ème}$ objectif.

D. La méthode ε-contrainte :

Cette méthode est basée sur la minimisation d'un objectif f_i en considérant que les autres objectifs f_j avec $j \neq i$ doivent être inférieurs à une valeur ε_j. En général, l'objectif choisi est celui que le décideur souhaite optimiser en priorité :

$$\text{Minimiser } f_i(x) \text{ avec } : f_j(x) \leq \varepsilon_j \forall j \neq i$$

De cette manière, un problème simple objectif sous contraintes peut être résolu. Le décideur peut ensuite répéter ce processus sur un objectif différent jusqu'à ce qu'il trouve une solution satisfaisante.

II.9.4.2 Les méthodes non agrégées et non Pareto

Certaines méthodes n'utilisent aucun des deux concepts précédents. Alors que l'agrégation ou l'utilisation de la dominance de Pareto traitent les objectifs simultanément, en général, les méthodes dites non agrégées et non Pareto possèdent un processus de recherche qui traite séparément les objectifs **[16]**.

a) Vector Evaluated Genetic Algorithm (VEGA)

La seule différence avec un algorithme génétique simple est la manière dont s'effectue la sélection. Si nous avons k objectifs et une population de n individus, une sélection de n/k individus est effectuée pour chaque objectif. Ainsi k sous-populations vont être créées, chacune d'entre elles contenant les n/k meilleurs individus pour un objectif particulier.

Les k sous-populations sont ensuite mélangées afin d'obtenir une nouvelle population de taille n. Le processus se termine par l'application des opérateurs génétiques de modification (croisement et mutation).

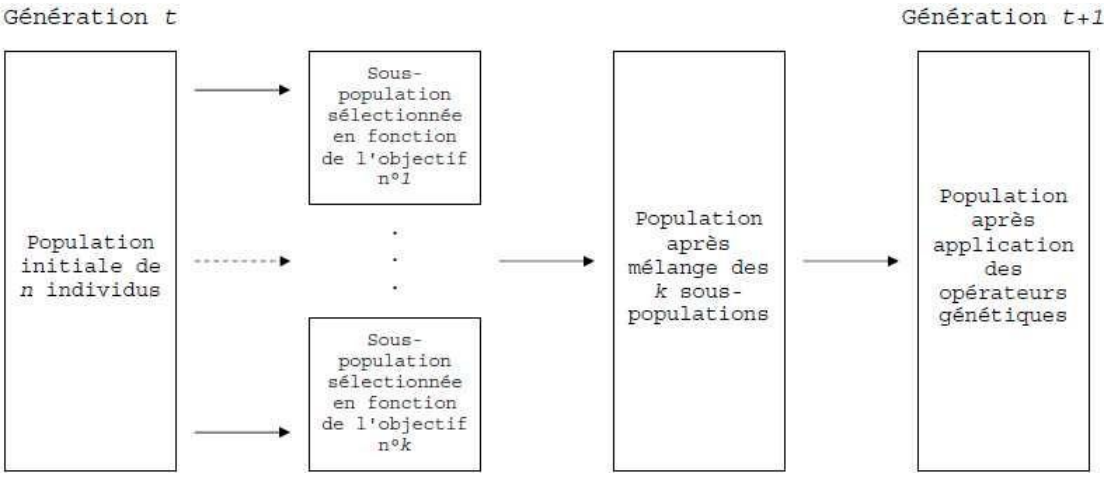

Figure II.12 Schéma de fonctionnement de VEGA **[16].**

b) La méthode lexicographique :

Les objectifs sont préalablement rangés par ordre d'importance par le décideur. Ensuite, l'optimum est obtenu en minimisant tout d'abord la fonction objectif la plus importante puis la deuxième et ainsi de suite.

L'expression mathématique du problème Soient les fonctions objectifs f_i avec i = 1, …, k, supposons un ordre tel que $f_1 > f_2 > \ldots > f_k$ Il faut :

$$\text{Minimiser} f_1(x) \text{Avec } g_j(x) \text{ satisfait} \forall j=1,\ldots,m$$

-Soit x1*, la meilleure solution trouvée avec f*1 = f1(x1*). F1* devient alors une nouvelle contrainte, l'expression du nouveau problème est donc :

$$\text{Minimiser} f_2(x) \text{Avec } g_j(x) \text{ satisfait } \forall j=1,\ldots,m \text{ et} f_1(x)=f^*_1$$

-Soit x2* la solution de ce problème. Le $i^{ème}$ problème sera le suivant :

Minimiser $f_i(x)$Avec $g_j(x)$ satisfait $\forall j=1,\ldots,m$ **et** $f_1(x)=f^*_1$, $f_2(x)=f^*_2$, $f_3(x)=f^*_3$, …, $f_{i-1}(x)=f^*_{i-1}$

La procédure est répétée jusqu'à ce que tous les objectifs soient traités. La solution obtenue à l'étape k sera la solution du problème.**[16].**

II.9.4.3 Les méthodes fondées sur Pareto

Ces méthodes sont fondées sur la notion de dominance au sens de Pareto qui privilégie une recherche satisfaisante au mieux tous les objectifs **[16]**.

Elles suggèrent d'utiliser le concept d'optimalité de Pareto pour respecter l'intégralité de chaque critère car il refuse de comparer a priori les valeurs de différents critères. L'utilisation d'une sélection basée sur la notion de dominance de Pareto va faire converger la population vers un

Ensemble de solutions efficaces.

a) Optimum de Pareto :

Au XIX^{ème} siècle, Vilfredo Pareto, un sociologue et économiste italien, formule le concept suivant : dans un problème multi objectif, il existe un équilibre tel que l'on ne peut pas améliorer un critère sans détériorer au moins un des autres critères.

Cette équilibre a été appelé optimum de Pareto. Un point x est dit Pareto- optimal s'il n'est dominé par aucun autre point appartenant à E (L'ensemble des actions réalisables). Ces points sont également appelés solutions non inférieures ou non dominées.

b) Notion de dominance

La notion de dominance consiste à attribuer un rang à chaque individu en favorisant la sélection des individus de rang le plus élevé pour rechercher une meilleure solution.
Lorsqu'on résoudre un problème d'optimisation multi-objectif, on obtient une multitude de solutions. Seul un nombre restreint de ces solutions va nous intéresser. Pour qu'une solution soit intéressante, il faut qu'il existe une relation de dominance entre la solution considérée et les autres solutions, dans le sens suivant :

c) La relation de dominance :

On dit que le vecteur x domine le vecteur x' si :

- x est au moins aussi bon que x' dans tous les objectifs.
- x est strictement meilleur que x' dans au moins un objectif.

C'est-à-dire, un point $x \in E$ **domine** $x' \in E$ si :

$$f_i(x) \leq f_i(x') \text{avec :au moins un i tel que } f_i(x) < f_i(x')$$

Dans la figure II.13 les points 1,3 et 5 ne sont dominés par aucun autre. Alors que le point 2 est dominé par le point 1, et que le point 4 est dominé par les points 3 et 5.

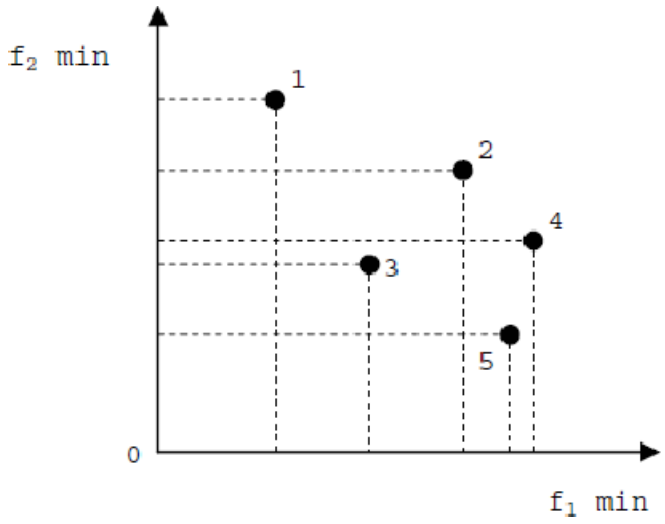

Figure II.13 : Exemple de dominance **[16].**

d) Frontière de Pareto :

La frontière de Pareto est l'ensemble de tous les points Pareto-optimaux.

La figure II.14 présente pour un problème à deux objectifs les quatre frontières de Pareto en fonction du désir de l'utilisateur de minimiser ou maximiser les objectifs.

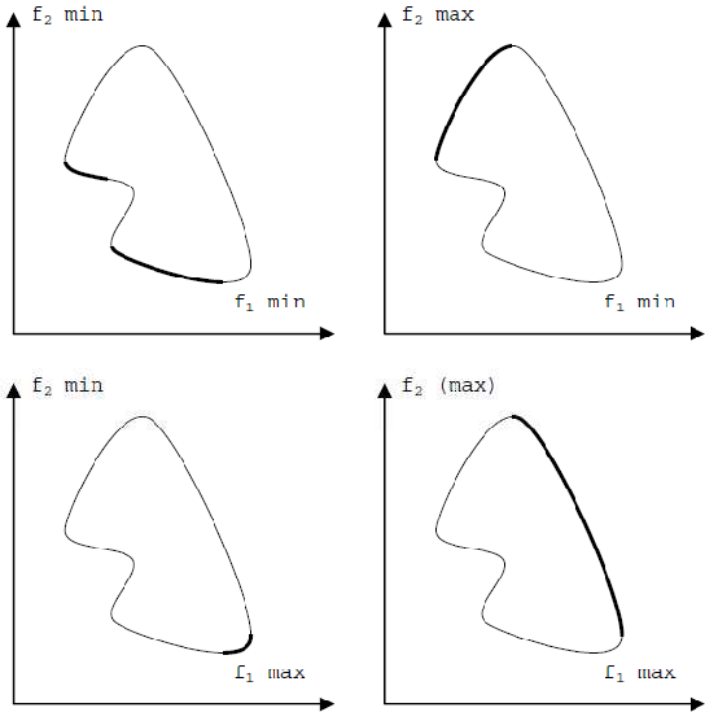

Figure II.14 Exemples de frontière de Pareto **[16].**

e) La représentation de la surface de compromis :

Toutes les représentations de la surface de compromis, pour un même problème, ne sont pas équivalentes. En effet, la représentation idéale de la surface de compromis devra être constituée de points solution de notre problème répartis de manière uniforme sur la surface de compromis.

Dans le cas où les points représentant la surface de compromis ne sont pas répartis de manière uniforme, l'utilisateur n'aura alors pas en sa possession un ensemble de solutions très utile **[19].**

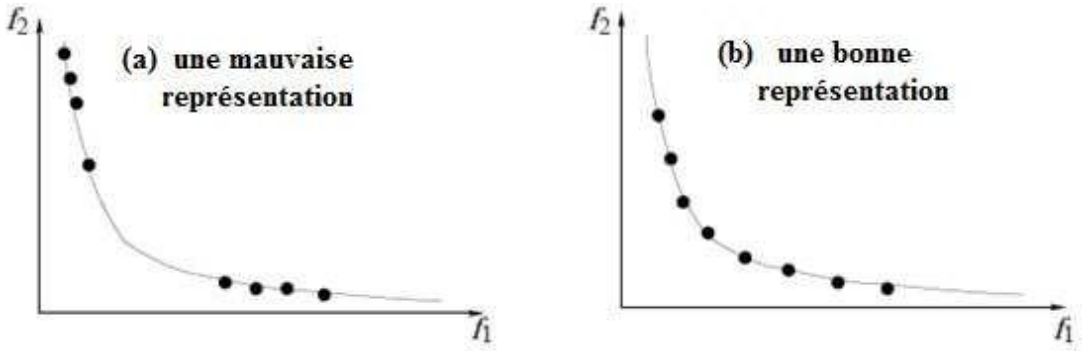

Figure II.15 : La représentation de la surface de compromis **[20]**.

II.9.4.4 Techniques non élitistes

1. Multiple Objective Genetic Algorithm (MOGA) :

En 1993 Fonseca et Fleming ont proposé une méthode dans laquelle chaque individu de la population est rangé en fonction du nombre d'individus qui le dominent. Elle utilise la relation de dominance pour déterminer l'efficacité d'un individu.

Ensuite, ils utilisent une fonction de notation permettant de prendre en compte le rang de l'individu et le nombre d'individus ayant même rang.

Soit un individu xi à la génération t, dominé par $p_i(t)$ individus. Le rang de cet individu est :

$\text{Rang}(x_i, t) = 1+p_i(t)$

A tous les individus non dominés on affecte le rang 1. Les individus dominés se voient donc affectés à un rang important. Pour le calcul de l'efficacité, on peut suivre les étapes suivantes :

1. Classer les individus en fonction de leur rang.

2. Affecter une efficacité à un individu en interpolant à partir du meilleur (rang 1) jusqu'au plus mauvais (rang n), en utilisant une fonction f (rang)**[19].**

La fonction d'assignation du rang de Pareto est représenté dans l'algorithme suivant, la variable N désigne le nombre de points de l'ensemble sur lequel on effectue les comparaisons.

Algorithme 2.1 : Algorithme d'assignation du rang de Pareto

```
RangCourant = 1
m = N
while N ≠ 0 do
For i = 1 to m do
If Xi est Non dominé
Rang(Xi, t) = RangCourant
End For
For i = 1 to m do

If Rang(Xi, t) = RangCourant
Ranger Xi dans une population temporaire
N = N −1

End For
RangCourant = RangCourant + 1
m = N
```

2. NSGA (Non dominated Sorting Genetic Algorithm):

Dans cette méthode proposée par Srivinas et Deb 1993, le calcul de fitness s'effectue en séparant la population en plusieurs groupes en fonction du degré de domination au sens de Pareto de chaque individu :

1. Dans la population entière, on recherche les individus non dominés. Ces dernier constituent la première frontière de Pareto.
2. On leur attribue une valeur de fitness factice, cette valeur est supposée donner une chance égale de reproduction à tous ces individus. Cependant pour maintenir la diversité dans la population, il est nécessaire d'appliquer une fonction de partage sur cette valeur.
3. Ce premier groupe d'individu est ensuite supprimé de la population.
4. On recommence cette procédure pour déterminer la seconde frontière de Pareto. La valeur factice de fitness attribuée à ce second groupe est inférieure à la plus petite fitness après application de la fonction de partage sur le premier groupe. Ce mécanisme est répété jusqu'à ce que l'on ait traité tous les individus de la population.

L'algorithme se déroule en suite comme un algorithme génétique classique.

3. Niched Pareto Genetic Algorithm(NPGA):

Cette méthode proposée par Horn et Napfliotis en 1993 utilise un tournoi basé sur la notion de dominance de Pareto. Elle compare deux individus pris au hasard avec une sous population de taille t_{dom} également choisie au hasard. Si un seul de ces deux individus domine le sous-groupe, il est alors positionné dans la population suivante.

Le paramètre t_{dom} permet d'exercer une pression variable sur la population et ainsi d'augmenter ou de diminuer la convergence de l'algorithme. Par exemple, si t_{dom}= 2 la convergence sera très lente.

A travers leurs expérimentations (Horn and Napfliotis) établissent le constat suivant :

- Si $t_{dom} \approx$ 1% de N (nombre d'individus de la population), il y a trop de solutions dominées. Cette valeur n'apporte pas assez d'élitisme dans la sélection, d'où un grand nombre de solutions non dominées qui subsistent dans la population.
- Si $t_{dom} \approx$ 10% de N, une bonne distribution des individus est obtenue. Cette valeur apporte un bon compromis entre élitisme et représentativité statistique de la population. Elle permet non seulement d'exercer une pression suffisante sur la sélection des individus, mais également d'éviter une convergence non désirée due à une sélection effectuée sur un échantillon non représentatif de la population.
- Si $t_{dom} >>$ 20% de N, il y a une convergence prématurée, car la pression est trop importante lors de la sélection**[16].**

II.9.4.5 Techniques élitistes

Les approches que nous avons vu sont dites non élitistes car :

- Elles ne conservent pas les individus Pareto-optimaux trouvés au cours du temps.
- Elles maintiennent difficilement la diversité sur la frontière de Pareto.
- La convergence des solutions vers la frontière de Pareto est lente.
- Pour résoudre les difficultés ci-dessus, de nouvelles techniques ont été appliquées.
- Introduction d'une population externe ou archive permettant de stocker les individus Pareto-optimaux.
- Utilisation des techniques pour répartir efficacement les solutions sur la frontière de Pareto.
- Préférence pour les solutions non dominées.

1. Strength Pareto Evolutionary Algorithm (SPEA):

En 1998 Zitzler et Thiele proposent une nouvelle méthode d'optimisation multi-objectif qui possède les caractéristiques :

- Utilisation du concept de Pareto pour comparer les solutions.
- Un ensemble de solutions Pareto-optimales est maintenu dans une population externe appelée archive.
- La fitness de chaque individu est calculée par rapport aux solutions stockées dans l'archive.
- Toutes les solutions de l'archive participent à la sélection.

➢ **Algorithme :**

Le passage d'une génération à une autre commence par la mise à jour de l'archive. Tous les individus non dominés sont copiés dans l'archive et les individus dominés déjà présents sont supprimés. Ensuite la fitness de chaque individu est mise à jour avant d'effectuer la sélection en utilisant les deux ensembles. Pour terminer, on applique les opérateurs génétiques de modification.

2. Pareto Envelope based Selection Algorithm (PESA) :

La méthode PESA a été également proposée par Knowles et Corne. Elle définit un paramètre appelé squeeze_factor qui représente la mesure d'encombrement d'une zone de l'espace. PESA est une méthode basée sur les algorithmes génétiques. Elle définit deux paramètres concernant la taille des populations d'individus : P_I (taille de la population interne) et P_E (taille de la population externe ou archive)**[12].**

➢ **Algorithme :**

a) Génération aléatoire et évaluation de la population P_I, et initialisation de $P_E = \emptyset$.

b) Transfert de tous les individus non dominés de PI dans PE.

c) Si le critère d'arrêt est réalisé **alors**

On retourne P_E comme ensemble de solutions.

Sinon on supprime tous les individus de PI et on recrée PI de la façon suivante :

On sélectionne deux parents dans P_E avec la probabilité p_c, on produit un enfant par croisement puis on le mute. Avec la probabilité $(1-p_c)$, on sélectionne un parent et on le mute pour produire un autre enfant.

d) On recommence au b.

Une solution courante de P_I peut entrer dans l'archive P_E si elle est non dominée dans P_I et si elle est non dominée dans P_E. Une fois, la solution insérée dans l'archive, on supprime tous les

membres de l'archive qu'elle domine. Si l'ajout crée un dépassement de capacité de P_E alors le membre de l'archive ayant le paramètre squeeze_factor le plus élevé est supprimé **[16].**

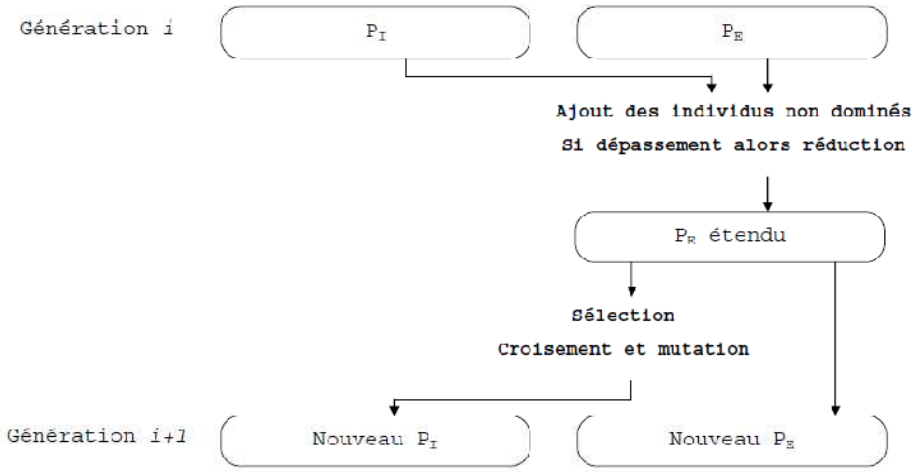

Figure II.16 : Schéma de fonctionnement de PESA.

Le paramètre squeeze_factor est égal au nombre d'individus qui appartiennent au même hyper cube. Il est utilisé comme fitness des individus qui appartiennent à cette zone.

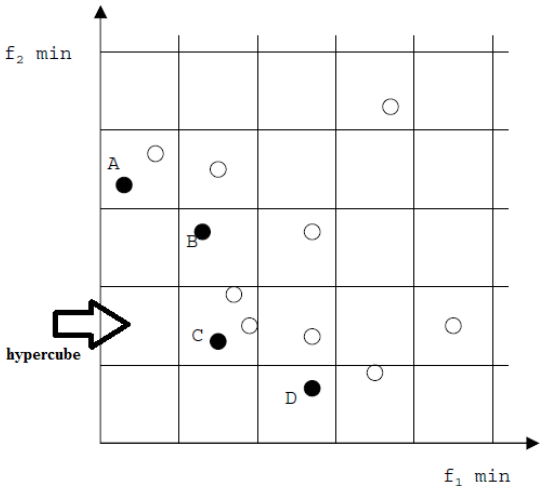

Figure II.17 : Mesure du squeeze_factor de PESA

Par exemple, dans la figure II.17 , les points B et D ont un squeeze_factor égal à 1 et C a un squeeze_factor égal à 3. Ce paramètre est utilisé pour la sélection ainsi que pour la mise à jour de l'archive. Cette méthode permet d'une bonne répartition des individus une dans l'espace d'état.

II.10 Conclusion

Un processus d'optimisation multi-objectif au sens de Pareto doit résoudre les deux tâches suivantes :

- Guider le processus de recherche vers la frontière de Pareto,
- Maintenir une diversité des solutions pour assurer une bonne répartition sur la frontière de Pareto.

L'accomplissement de ces tâches est très délicat car les difficultés rencontrées dans un problème multi-objectif sont identiques à celles d'un problème simple objectif mais elles sont amplifiées par la présence d'objectifs dépendants les uns des autres.

CHAPITRE III : Contribution et Implémentation

III.1.Introduction :

Dans ce chapitre nous mettons en œuvre les principes que nous avons vus précédemment. Premièrement on commence par rappeler la technique de résolution des problèmes d'optimisation par introduction des algorithmes génétique multi-objectif la technique d'agrégation (La moyenne pondérée) et la notion l'échantillonnage de la population initial pour pouvoir évaluer les résultats.

III.2.Echantillonnage :

L'échantillonnage est la phase qui consiste a sélectionner les individus que l'on souhaite interroger au sein de la population de base.[20]

III.3. Méthodes d'échantillonnage :

L'échantillonnage peut se faire avec ou sans remise et une population peut être considérée comme finie ou infinie. Une population finie dans laquelle on procède a un échantillonnage avec remise peut être théoriquement considérée comme infinie. Dans la pratique, il en va de même pour des populations finies mais de grandes tailles. Pour chaque distribution d'échantillonnage, on peut calculer une moyenne, un écart type, une variance…etc[20]

III.3.1. Méthodes probabilistes (Aléatoires) :

L'échantillonnage probabiliste repose sur un choix d'unités dans la population fait au hasard, ce n'est pas l'enquêteur qui choisit les unités, c'est la méthode utilisée pour la sélection qui le fait. Une des caractéristiques de cette méthode est que chaque unité de la population a une probabilité mesurable d'être choisie.

L'avantage de la méthode d'échantillonnage probabiliste est qu'elle permet de généraliser les résultats de l'échantillon a l'ensemble de la population en s'appuyant sur une théorie statistique reconnue.

Son seul inconvénient est qu'il faut posséder une liste de toutes les unités formant la Population avant de procéder a la sélection de l'échantillon.

Voici les quatre types d'échantillonnage probabiliste que l'on peut effectuer :[20]

III.3.2. Echantillonnage aléatoire simple :

Un échantillon aléatoire simple est un échantillon sélectionnée de manière a ce que chaque échantillon possible de taille "**n**" ait la même probabilité d'être sélectionne, On prélevé dans la population des individus au hasard, tous les individus

ont la même probabilité d'être prélevés, et ils le sont indépendamment les uns des autres.[20]

III.3.3. Echantillonnage aléatoire stratifie :

On suppose que la population soit stratifiée, constituée de sous-populations homogènes, les strates. (ex : stratification par tranche d'Age). Dans chaque strate, on fait un échantillonnage aléatoire simple, de taille proportionnelle à la taille de strate dans la population (échantillon représentatif). Les individus de la population n'ont pas tous la même probabilité d'être tires.[20]

III.3.4. Echantillonnage aléatoire par grappe :

On tire au hasard des grappes ou familles d'individus, et on examine tous les individus de la grappe (ex: on tire des immeubles puis on interroge tous les habitants). La méthode est d'autant meilleure que les grappes se ressemblent et que les individus d'une même grappe sont différents, contrairement aux strates. [20]

III.3.5. Echantillonnage aléatoire systématique :

Dans certaines situations, spécialement lorsque les populations sont importantes, il est couteux (en temps) de sélectionner un échantillon aléatoire simple en trouvant tout d'abord un nombre aléatoire et ensuite en cherchant dans la liste de la population l'élément correspondant .[20]

III.3.6. Méthodes non probabilistes (Raisonnées ou empirique) :

L'échantillonnage non probabiliste repose sur un choix arbitraire des unités, c'est l'enquêteur qui choisit les unités et non le hasard. En ce sens, il serait donc aventureux de généraliser les résultats obtenus pour l'échantillon à toute la population. Malgré cela, ces méthodes sont souvent utilisées dans certaines disciplines. En voici quelques-unes :[20]

1. **Echantillonnage par quota :**

Lorsque le chercheur veut reproduire les caractéristiques d'une population (ex. Âge, sexe, revenus, etc.) dans son échantillon.

2. **Echantillonnage de convenance (de commodité) :**

   Cas ou les unités d'échantillonnage sont faciles à rejoindre, disponibles et généralement facile à convaincre.

3. **Echantillonnage selon le jugement :**

Le chercheur juge que l'échantillon va lui permettre d'atteindre les objectifs

de la recherche.

4. **Echantillonnage boule de neige :**

Utile dans le cas de la rareté des unités d'échantillonnage ou de l'absence d'un cadre d'échantillonnage valide. On demande à un répondant de nous référer a un autre qui présente les mêmes caractéristiques que les siennes, et ainsi de suite…

III.4.Outils utilisés :

Pour réaliser ce travail on a utilisé :

III.4.1. Outils matériels :

Un ordinateur potable caractérise par :

1. Un processeur (CPU) : Core(MT)i3 Duo 2.10 Ghz
2. Une mémoire (RAM) : 04 Go

III.4.2. Outils logiciels :

La version simulation utiliser est Matlab version R2013a :

MATLAB (MATrix LABoratory) est un environnement puissant, complet et facile a utiliser destine au calcul scientifique. Il apporte aux ingénieurs, chercheurs eta tout scientifique un système interactif intégrant calcul numérique et visualisation. C'est un environnement performant, ouvert et programmable.

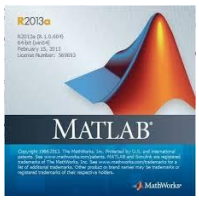

Figure III.1 : logo Matlab (version R2013a)

III.5. Organigramme :

Dans cet organigramme en va présenter l'aide de notre travail de manière général :

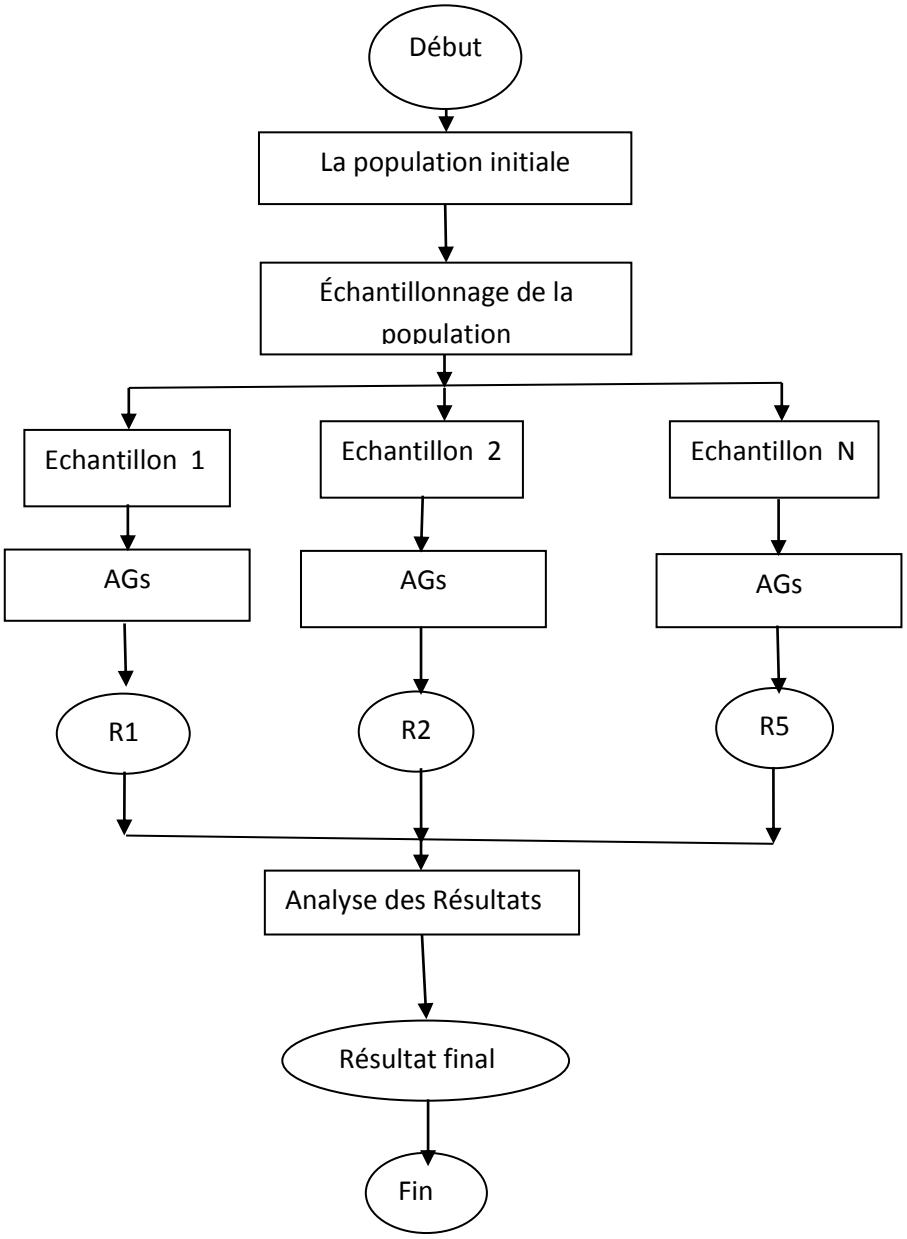

Figure III.2 organigramme

III .6 Algorithme

1) Initialiser la population initiale P.
2) Echantillonne la population initiale en des échantillons égaux
3) Evaluer échantillon i
4) Tant Que (condition d'arrêt 1)

a)Pi ' = Sélection des Parents dans échantillon i

b) Pi ' = Appliquer Opérateur de Croisement sur Pi ‘

c) Pi ' = Appliquer Opérateur de Mutation sur Pi '

d) échantillon i = Remplacer les Anciens de P par leurs Descendants de Pi ‘

e) Evaluer P FinTantQue

5) échantillon i = échantillon i+1 aller à(3)

III.7. Fonctions objectifs (fitness).

Sert à évaluer la précision de la solution du problème. Les fonctions objectives Utilisées dans notre étude :

- Fonction objectif sans contrainte.
-

$$\begin{cases} f_1(x) = (x+2)^2 - 10 \\ f_2(x) = (x-2)^2 + 20 \end{cases} \qquad x \in R$$
$$-1 \leq x \leq 1$$

- Fonction objectif avec contrainte.

$$\begin{cases} f_1(x_1, x_2) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2 \\ f_2(x_1, x_2) = 9x_1 - (x_2 - 1)^2 \end{cases} \qquad x_1, x_2 \in R^2$$

Avec les contraintes $x_1^2 + x_2^2 \leq 225$

$$x_1 - 3x_2 \geq -10$$
$$-2 \leq x_1, x_2 \leq 2$$

III.8. Méthode de résolution des problèmes d’optimisation multi-objectif

1. Les méthodes agrégées (La moyenne pondérée)

Cette méthode transforme un problème multi objectif en un problème simple objectif **[16]**.

$$\min \sum_{i=1}^{k} w_i f_i(x) \qquad \text{avec } w_i \geq 0$$

w_i représente le poids affecté au critère i et :

$$\sum_{i=1}^{k} w_i = 1$$

- Fonction objectif sans contrainte.

$$f(x) = a((x+2)^2 - 10) + b((x-2)^2 + 20) \quad \text{avec } a + b = 1$$

- Fonction objectif avec contrainte.

$f(x_1, x_2) = a((x_1 - 2)^2 + (x_2 - 1)^2 + 2) + b(9x_1 - (x_2 - 1)^2)$ avec a+b=1

Avec les contraintes $x_1^2 + x_2^2 \leq 225$

$$x_1 - 3x_2 \geq -10$$

$$-2 \leq x_1, x_2 \leq 2$$

2. Sélection (Méthode de la loterie biaisée (roulette))

Le principe de roulette peut être simulé par l'algorithme suivant :

Algorithme : Algorithme de Sélection par roulette**.**

1. On calcule la somme S1 de toutes les fonctions d'évaluation d'une population;
2. On génère un nombre r entre 0 et S1;
3. On calcule ensuite une somme S2 des évaluations en s'arrêtant dès que r est dépassé;
4. Le dernier individu dont la fonction d'évaluation vient d'être ajoutée est Sélectionné.

3. Croisement et mutation :

Les phénomènes de croisement et mutation est déjà expliqué dans le chapitre précédant.

III.9. Implémentation

III.9.1 Programme principale

```
clc;
close all;
clear all;
%************Partier declaration****************

Tail_Popul=200;    %***la taille de le Population
Tail_Echl=40;      %***la taille d'échontillon
gene= 5;           %***nombre de gene


solution=zeros(5,1);

Novel_Popul   = zeros(Tail_Popul ,gene); %***population intermédiaire
Moves_Popul  = zeros(Tail_Popul ,gene); %***Mouvese  population
Popul_Init= randn(Tail_Popul ,gene);  %**chargement de la population Initiale
fav  = zeros(Tail_Echl,1);%***vecteur max de résultat de calcule de fonction objectif

mov_fav =zeros(Tail_Echl,1); %vecteur min de résultat de calcule de fonction objectif

Prob_Crois  = 0.8;  %***probabilité de croisement
Prob_Mut  = 0.01;   %***probabilité de mutation

cromossomo_aux = zeros(1,gene);

Moves_cromos = zeros(1,gene); %***Movese cromosome

Nb_G = 5;   %***nombre de generation
Nb=1;

MV_x= 0;

estagnacao  = 0;
VectMV_x    = zeros(Nb_G,1);


% **********début de  programme***************

 a = rand(1,1); %***le choix des poids affecté au critère
b = 1-a;

%*************************************************************************
j = 1;

for k =1 :5

while(1)

if (Nb  > Nb_G)%***condition d'arrêt sur nbr de generation
break;
end
```

```
%***la fonction agrégation des fonction objectif********
fav  = zeros(Tail_Echl,1);

for i = j : Tail_Echl

y = Popul_Init(i,:);

      fav(i) = max(a*((y+2).^2 -10)+b*((y-2).^2 + 20)) ;%***max évaluation de fonction objectif

      mov_fav(i)=min(a*((y+2).^2 -10)+b*((y-2).^2 + 20)) ; %***min évaluation de fonction objectif

end

   [M_x, M_fav] = max(fav);%***sauvegarder de la valeur me fav max et la valeur x


%************************************************

if (Nb  ~= 1)
      MV_x =( MV_xi);
end

   [MV_xi,MV_F]=min(fav);

   VectMV_x(Nb)=MV_xi;

if ( MV_xi == MV_x)
      estagnacao = estagnacao + 1;
end

if(estagnacao> 5) %***une 2 condition d'arrêt si le moves gène  apparece 5 successive
break;
end

%  ****calcule de porsontage pour faire la roulet

somafav = sum(fav);  %*** la somme de Fi
   adr = zeros(Tail_Echl,1);


for i = j : Tail_Echl %***la devision de Fi/sommeFi
adr(i) = fav(i)/(somafav);
end

  rolette=zeros(Tail_Echl,1);

   rolette(j) = adr(j);

for i= j+1:Tail_Echl
      rolette(i) = rolette(i-1)+adr(i);
end

%***la selection ********

 selecion = zeros(Tail_Echl,1);
```

```
for r=j:Tail_Echl

       bola    = rand(1);
       position = j;
       flag    = 0;

while(1)
if ( bola <= rolette(position))

              selecion(r) = position;
              flag = 1;
end
if (flag == 1)
break;
end
    position = position + 1;% la probabilité la plus grande
end
end
 qtselecion  = 0;
 cjselecion =[];

 cjselecion = hist(selecion,Tail_Popul );

 indice = j;

for i = j:Tail_Echl

if (cjselecion (i) ~= 0)
          nobt = cjselecion (i);

for c = 1: nobt
             Novel_Popul(indice,:) =  Popul_Init(selecion(i),:);
             indice = indice + 1;
end
end
end

% ****croisement***************

vezes = 0.8 * Tail_Echl;

cromossomo_aux = zeros(1,gene);
qtd_trocas     = 0;
for r=j:vezes
       prob = rand*1;
if (prob <= Prob_Crois )
          qtd_trocas = qtd_trocas + 1;
          corte1 = ceil((rand*gene - 1)+1);
          corte2 = ceil((rand*gene - 1)+1);
if (corte1 > corte2)
             aux = corte1;
corte1 = corte2;
             corte2 = aux;
end

          individuo1 = ceil((rand*Tail_Echl-1)+1);
          individuo2 = ceil((rand*Tail_Echl-1)+1);
```

```
cromossomo_aux  =
(Novel_Popul(individuo1,corte1:corte2)+Novel_Popul(individuo2,corte1:corte2))/2;
Novel_Popul(individuo1,corte1:corte2)   = cromossomo_aux;
Novel_Popul(individuo2,corte1:corte2)   = cromossomo_aux;
end
end

%***mutation ***

   prob = rand*1;

if (prob <= Prob_Mut )
for i= j:Prob_Mut *Tail_Echl*gene
l = ceil(rand*Tail_Echl-1)+1;
c = ceil(rand*gene - 1)+1;
          Tail_Echl(l,c) = rand(1);
end
end

   Popul_Init = Tail_Echl;
   Nb      = Nb  + 1;
   disp(Nb );
end

%**** la représentation des figures de chaque échantillon
figure%
 plot(fav,'.g');%***
hold on;
plot(mov_fav,'.b');
hold on;
plot(max(fav),'.r');
xlabel('Population ');
ylabel('Evaluation de la Fonction objectif ');

%*****l'affichage de max de l'échantillon i

fprintf('%f \r \n',max(fav));

%****réinitialisation  des paramètre de la nouvelle échantillon

j=Tail_Echl+1;
Tail_Echl=Tail_Echl+40;

fav  = zeros(Tail_Echl,1);
mov_fav= zeros(Tail_Echl,1);

roleta=zeros(Tail_Echl,1);
adr = zeros(Tail_Echl,1);
MV_x= 0;

estagnacao  = 0;

Nb       = 1;end
```

III.9.2 Résultat et discutions

En a obtenue appré l'exécution de programme ;

Population initial :

	1	2	3	4	5	6	7	8	9	10
1	0.3648	0.2674	0.4068	-0.6986	-0.7382	1.3907	0.3216	-0.3635	0.2067	1.1418
2	0.7295	0.5349	0.8136	-0.6986	-0.3691	0.6954	0.3216	-0.3635	0.4134	2.2836
3	0.7295	0.5349	0.8136	-0.6986	-0.7382	1.3907	0.3216	-0.3635	0.4134	2.2836
4	0	0	0	0	-0.2461	0.4636	0.1072	-0.1212	0.1378	0
5	0	0	0	0	0	0	0	0	0.1034	0
6	0	0	0	0	-0.2461	0.4636	0.1072	-0.1212	0.1378	0
7	0	0	0	0	-0.3691	0.6954	0	0	0	0
8	0.3648	0.2674	0.4068	0	0	0	0	0	0.1034	1.1418
9	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0

Tableau III.1 Population initial

Evaluation mex fonction objectif dans chaque échantillon :

	1
1	11.0182
2	2.8241
3	0.8246
4	0.8246
5	0.8246

Tableau III.2 Résultats final d'évaluation d'échantillon Population initial

Remarque : c'est là ou le décideur décide la meilleur résultat pour son système.

Figures d’échantillons.

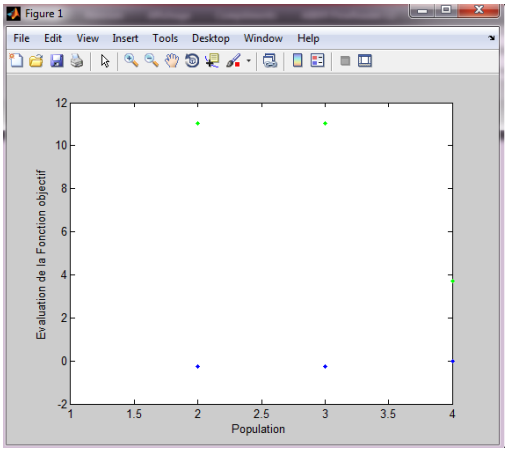

Figure1 : R1= 11.0182

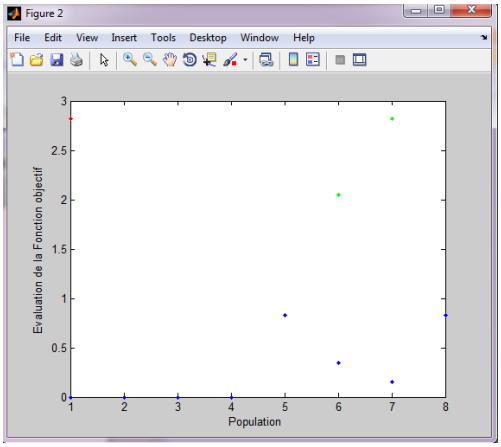

Figure2 : R2= 2.8241

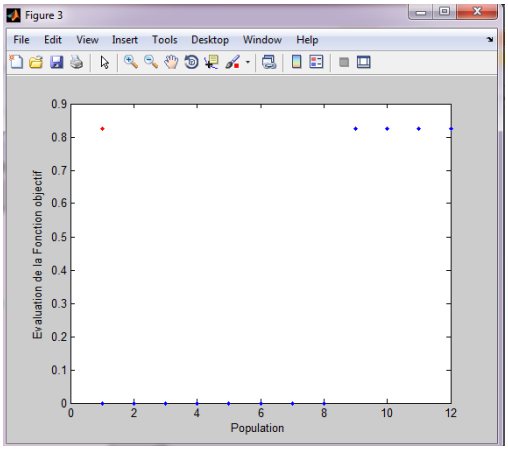

Figure3 : R3= 0.8241

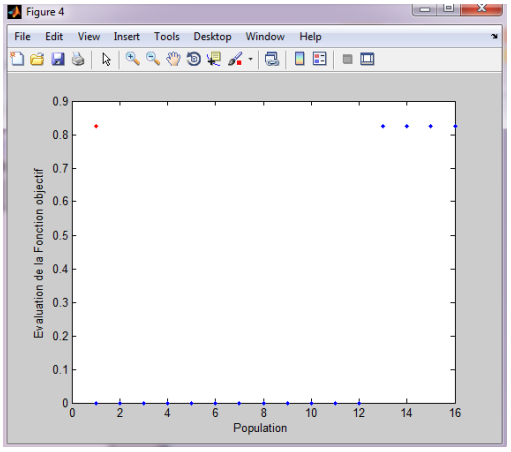

Figure4 : R4= 0.8241

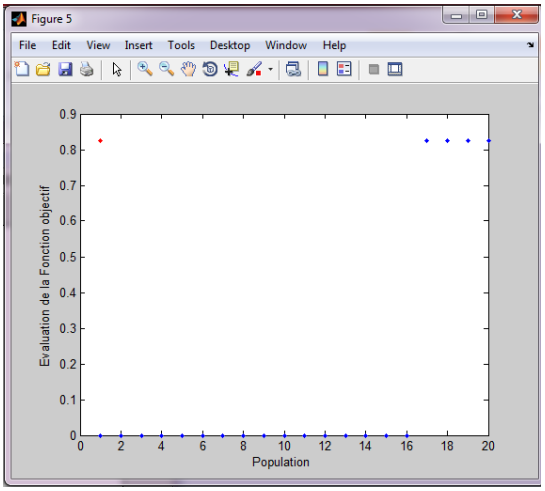

Figure5 : R5= 0.8241

III.10 Conclusion

Afin de répondre à l'objectif de concevoir un algorithme d'optimisation combinatoire multi-objectif efficace, en temps de calculs (raisonnable), en qualité de solutions produites et permettant de traiter des problèmes de grande taille ;
Nous avons proposé une méthode d'agrégation /échantillonnage, que nous avons appliquée au problème général.

Conclusion générale

Les algorithmes évolutionnaires semblent être spécialement utiles en optimisation multi-objective car ils sont capables de déterminer plusieurs solutions optimales en une seule exécution et peuvent exploiter les similarités des solutions par les opérateurs de variation. Les algorithmes génétiques se portent bien à l'optimisation multi-objectif.

Pour les problèmes d'optimisation multi-objectif sous contraintes, il faut satisfaire un ensemble de contraintes tout en optimisant plusieurs fonctions objectives. En conséquence, même avec un petit nombre de variables et d'objectifs, le problème ne peut pas être traité simplement avec un algorithme classique. En effet l'optimisation d'un problème multi-objectif est souvent plus difficile que l'optimisation des problèmes mono-objectif car la solution optimale ne se réduit plus à un seul point, mais à un ensemble de points.

Nous pouvons dire que les algorithmes génétiques présentent une classe de méthodes adaptables à une grande variété de problèmes d'optimisation combinatoire et mènent à des résultats pertinents. Mais il existe assez peu de contribution permettant de comprendre la raison de cette efficacité.

L'efficacité des algorithmes génétiques dépend fortement du réglage des différents paramètres caractérisant ces algorithmes, et qui sont parfois difficiles à déterminer. Des paramètres comme la taille de la population, le nombre maximal des générations, la probabilité de mutation, et la probabilité de croisement.

Le succès des algorithmes génétiques dépend aussi de la manière du codage des individus. Dans les problèmes combinatoires, ce codage est souvent suggéré par la nature même du problème, ce qui peut induire de meilleurs résultats.

Au final, l'ensemble des expérimentations effectuées souligne donc l'efficacité des AGs comme outil de résolution. Tous cependant mettent en avant une limite : si les AGs convergent vers une solution optimale rien ne permet de dire. En outre, les AGs peuvent rester longtemps proche de la solution optimale sans l'atteindre. C'est la raison pour laquelle de nombreuses méthodes dites hybrides, combinant AG et d'autres méthodes, sont de plus en plus utilisées. Enfin, la durée de calcul (temps CPU) peut être longue.

Comme perspectives et pour mieux améliorer notre travail présenté dans ce mémoire on peut proposer :

Il est possible d'améliorer les résultats par la mise en œuvre d'une ou plusieurs heuristiques pour générer la population initiale selon le problème étudié pour générer des solutions faisables.

Lorsque la complexité de l'algorithme est importante, son approche distributive se prête très bien aux calculs en parallèle, ce qui permet de contourner le problème. Puisque les algorithmes génétiques apportent cependant une solution acceptable. Néanmoins, il est possible de les améliorer assez efficacement en les combinant avec d'autres algorithmes distribue.

.

Références bibliographiques

Références bibliographiques :

[1] Adorio, E. P., &Diliman, U. (2005). Mvf-multivariate test functions library in c for unconstrained global optimization.

[2] Alba, E., &Dorronsoro, B. (2009). Cellular genetic algorithms (Vol. 42). Springer Science & Business Media,13-20

[3] Baker, J. E. (1985, July). Adaptive selection methods for genetic algorithms.In Proceedings of an International Conference on Genetic Algorithms and their applications (pp. 101-111).

[4] Baker, J. E. (1987, July). Reducing bias and inefficiency in the selection algorithm.In Proceedings of the second international conference on genetic algorithms (pp. 14-21).

[5] Black, P. E. (2007). big-O notation. Dictionary of Algorithms and Data Structures, 2007.

[6] Coello, C. A. C., & Lamont, G. B. (2004). Applications of multi-objective evolutionary algorithms (Vol. 1).World Scientific.

[7] Corne, D. W., Knowles, J. D., & Oates, M. J. (2000, January). The Pareto envelope-based selection algorithm for multiobjective optimization.In Parallel Problem Solving from Nature PPSN VI (pp. 839-848).Springer Berlin Heidelberg.

[8] Darwin, C. (1859). On the origins of species by means of natural selection.London: Murray, 247.

[9] Deb, K., &Agrawal, R. B. (1994). Simulated binary crossover for continuous search space.Complex Systems, 9(3), 1-15.

[10] Deb, K., &Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design.Computer Science and Informatics, 26, 30-45.

[11] Deb, K., Pratap, A., Agarwal, S., &Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. Evolutionary Computation, IEEE Transactions on, 6(2), 182-197.

[12] Deb, K., Thiele, L., Laumanns, M., &Zitzler, E. (2002, May). Scalable multi-objective optimization test problems. In Proceedings of the Congress on Evolutionary Computation (CEC-2002),(Honolulu, USA) (pp. 825-830). Proceedings of the Congress on Evolutionary Computation (CEC-2002),(Honolulu, USA).

[13] Eskandari, H., Geiger, C. D., & Lamont, G. B. (2007, January). FastPGA: A dynamic population sizing approach for solving expensive multiobjective

optimization problems. In Evolutionary Multi-Criterion Optimization (pp. 141-155). Springer Berlin Heidelberg.

[14] Fonseca, C. M., & Fleming, P. J. (1993, June). Genetic Algorithms for Multiobjective Optimization: Formulation Discussion and Generalization. In ICGA (Vol. 93, pp. 416-423).

[15] Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance.Journal of the American Statistical Association, 32(200), 675-701.

[16] Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning (1st ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[17] Goldberg, D. E. (1990). Real-coded genetic algorithms, virtual alphabets, and blocking.Urbana, 51, 61801.

[18] Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms.Foundations of genetic algorithms, 1, 69-93.

[19] Holland, J. (1975). Adaptation in artificial and natural systems.Ann Arbor: The University of Michigan Press.

[20] Support pedagogique de cours N° 02 :**Echantillonnag «** KHERRI Abdenacer »'ECOLE DES HAUTES ETUDES COMMERCIALES'

[21]https://fr.wikipedia.org/wiki/Optimisation_(math%C3%A9matiques)

[22]'Etude comparative des algorithmes génétiques multi-objectifs'2014/2015 MEMOIERE DE FIN DE CYCLE