Ministère de L'enseignement Supérieur et de la Recherche Scientifique

Université Dr.Yahia Farès de Médéa

Département de Mathématiques et Informatique



MÉMOIRE

Présenté en vue de l'obtention du

DIPLÔME DE MASTER EN MATHEMATIQUES

Spécialité :Recherche Opérationnelle

par

BOUDOUMA Mouna

SERIR Nessrine

L'OPTIMISATION MULTIOBJECTIF PAR LA MÉTHODE NSGA-II

Devant le jury :

Directeur :		
Mr K.TABLENNEHAS	M.A.A	Université de Médéa
So-directeur :		
Mr K.SEBAA	Professeur	Université de Médéa
Examinateur :		
Mme K.SERIER ABDALLAH	M.A.A	Université de Médéa
Mme S.KOUIDER AIAD	M.A.A	Université de Médéa

Remerciments

Louange à dieu, le miséricordieux, sans lui rien de tout cela n'aurait pu être.

Nous tenons tout d'abord ,à exprimer notre profond gratitude à notre promoteur monsieur TABLENNEHAS Kamel,Maître assistant à l'université Dr.Yahia Farès de Medea ,qui a assumé la direction de ce travail.

sa patince,sa disponibilité ,ses conciels et son aide constant qu'il nos apporté tout au long de ce travail.

Nous voudrons sincèrement exprimer nos plus vifs remerciements à notre co-promoteur monsieur SEBAA Karim,professeur à l'université Dr.Yahia Farès de Medea s'est toujours montrée à l'écoute et très disponible tout au long de la réalisation de ce travail,ainsi,pour son aide et le temps qu'il a bien voulu nos consacrer.

Nous adressons notre vifs remerciements à tous les enseignants qui, par leur enseignement, leur encouragement et leur aide, ont contribué à notre formation durant toutes notre études à l'université de Dr.Yahia Farès de Medea.

Dédicaces

Je remercie le Dieu pour m'avoir donné la force d'accomplir ce travail pour aller plus loin.
Je dédie ce travail à mes parents, ma mère pour ses encouragements et ses prières tout au long de mes études, mon père pour tous ce qu'il avait fait pour avoir ce résultat.
Je le dédie à mes frères et ma sœur, et je les remercie pour leurs encouragements et leurs aides ainsi que toute ma grande famille.
A tous mes amis sans citer les noms.
A mes collègues de la promotion 2020-2021 de post-graduation.
A tous ceux qui aiment Mouna et ceux qui Mouna aime.

MOUNA

Dédicaces

Je dédie ce modeste travail :

A mes très chers parents qui sont la source de mon éducation, mon savoir et mes principes.

A mes très chers frères et ma soeur pour leur soutien.

A tous les gens que j'aime et dont je n'ai pas cité les noms.

NESSRINE

Résumé

Les problèmes d'optimisation multiobjectifs ont toujours été des problèmes difficiles et brûlants dans le domaine de la recherche scientifique et technique. Les solutions traditionnelles présentent de nombreuses lacunes pour traiter des problèmes complexes tels que les grandes dimensions et la multimodalité. Pour résoudre ce problème, une variété d'algorithmes génétiques d'optimisation multiobjectifs a été développée. L'algorithme génétique de tri non dominé (NSGA) et son algorithme amélioré NSGA-II étudiés dans ce mémoire sont développés plus rapidement et optimisés.ce mémoire a systématiquement étudié et recherché les principes de base de l'algorithme, fait des tentatives audacieuses dans l'application de l'algorithme, élargi le champ d'application de l'algorithme et proposé une nouvelle simulation de fonction de solution La méthode de combinaison du problème. Les principaux travaux réalisés dans ce mémoire sont les suivants :

Le processus général et les principes de base de l'algorithme génétique sont systématiquement introduits Et l'algorithme génétique d'optimisation multiobjectif existant et certains de ses problèmes. Les principes de base et les lacunes de NSGA sont discutés, et la méthode de tri rapide non dominée proposée par l'algorithme amélioré NSGA-II.

Mots-clés : Optimisation multiobjectif, algorithmes génétiques, NSGA-II.

Abstract

Mufti-objective optimization problems have always been difficult and burning problems in the field of scientific and technical research. Traditional solutions have many gaps to address complex problems such as large-scale and multimodality. To solve this problem, a variety of multi-objective genetic optimization algorithms has been developed. The uncontrolled sorting genetic algorithm (NSGA) and its improved NSGA-II algorithm studied in this memory are developed more quickly and optimized. this memory has systematically studied and researched the basic principles of the algorithm, made daring attempts in the application of the algorithm, broadened the scope of the algorithm and proposed a new simulation of the solution function. The main work carried out in this memory is as follows :

The general process and basic principles of the genetic algorithm are systematically introduced And the existing multiobjective optimization genetic algorithm and some of its problems. The basic principles and shortcomings of NSGA are discussed, and the rapid sorting method not dominated proposed by the improved NSGA-II algorithm.

Keywords : Multiobjective optimization, genetic algorithms, NSGA-II.

ملخص

كانت مشاكل تحسين الأهداف المفتي صعبة ومشكلات حارقة في مجال البحث العلمي والتقني. فالحلول التقليدية لها فجوات عديدة لمعالجة المشاكل المعقدة مثل تعدد الطرائق على نطاق واسع. لحل هذه المشكلة، تم تطوير مجموعة متنوعة من خوارزميات التحسين الجيني متعددة الأهداف. يتم تطوير الخوارزمية الجينية للفرز غير الخاضع للتحكم (صجا) وخوارزمية صجاايي المحسنة التي تمت دراستها في هذه المذكرة بشكل أسرع وتحسينه. وقد قامت هذه المذكرة بدراسة المبادئ الأساسية للخوارزمية والبحث فيها بشكل منتظم، وجعلت محاولات جريئة في تطبيق الخوارزمية، ووسعت نطاق الخوارزمية واقترحت محاكاة جديدة لوظيفة الحل. العمل الرئيسي الذي يتم في هذه المذكرة هو كما يلي :
يتم بشكل منتظم إدخال العملية العامة والمبادئ الأساسية للخوارزمية الجينية، بالإضافة إلى تحسين الأداء متعدد الأهداف الحالي والخوارزمية الجينية وبعض المشاكل التي تواجهها. تتم مناقشة المبادئ الأساسية وأوجه القصور في صجا، ولا يتم فرض نظام الفرز السريع الذي لا تهيمن عليه خوارزمية صجاايي المحسنة.
الكلمات الافتتاحية : تحسين متعدد الأهداف، خوارزميات وراثية، صجاايي

Table des matières

Table des figures

1 OPTIMISATION MONO-OBJECTIF **4**

1.1 Définition d'un problème d'optimisation 4

1.1.1 Terminologie de la solution 6

1.1.2 Minimum et Maximum local 6

1.1.3 Minimum et Maximum global : 6

1.1.4 Notion de convexité : 7

1.1.5 Gradient et Matrice Hessien : 7

1.2 Classification générale des méthodes d'optimisation mono-objectif 8

2 OPTIMISATION MULTI-OBJECTIF **10**

2.1 Structure d'un problème d'optimisation multi-objectifs 10

2.2 Les variables d'un problème multi-objectif 14

2.3 Multiplicité des solutions 15

2.4 Résoudre un problème multi-objectif 16

2.5 Classification des problèmes multi-objectifs : 17

2.5.1 Approches Pareto 18

2.5.2 Optimalité de pareto 21

2.6 Surface de compromis 24

2.7 Efficacité (Ou Pareto optimale) : 30

2.7.1 Conditions nécessaires d'optimalités 31

2.7.2 Conditions suffisantes d'optimalité 32

2.8 Méthodes de résolution des problèmes multi-objectif : 32

2.8.1 Méthodes exactes 34

2.8.2 Les heuristiques 36

2.8.3 Approches non Pareto : 39

2.8.4 Approches Pareto : 40

2.9 Différents méthodes de résolutions des problèmes d'optimisation multi-objectifs . . 41

2.9.1 Méthodes scalaires 42

2.9.2 Méthodes interactives 42

2.9.3 Méthodes floues 43

2.9.4 Méthodes exploitant une méta-heuristique 44

2.9.5 Méthodes d'aide à la décision 44

3 MÉTA-HEURISTIQUE ET ALGORITHMES GÉNÉTIQUES 46

3.1 Définition de la simulation : 46
3.2 Heuristique 48
3.3 Méthodes de trajectoire 49
3.4 Méta-heuristique 50
3.4.1 Le Recuit Simulé (simulated annealing) 51
3.4.2 La recherche Tabou (Tabu Search) 53
3.5 Algorithmes génétiques 55
3.5.1 Fonctionnement des AGs 56
3.5.2 Représentation chromosomique(Codage) 57
3.5.3 Evaluation 59
3.5.4 Sélection 59
3.5.5 Opérateurs génétiques 59

4 MÉTHODE NSGA-II (A Fast Multiobjective Genetic Algorithm) 62

4.1 NSGA-I 62
4.1.1 NSGA-II 64
4.1.2 Stratégie d'amélioration 70
4.2 Les travaux de simulation 72
4.2.1 Sélection des paramètres 72
4.2.2 Simulation 73

Bibliographie 84

Table des figures

1.1 Classification générale des méthodes d'optimisation mono-objectif 8

2.1 Espace de recherche et espace réalisable 12

2.2 définition de E,F et f. 13

2.3 La présentation mathématique d'un problème d'optimisation multi-objectif. 13

2.4 Modes de résolution d'un problème Multi-objectif 17

2.5 Dominance de pareto ou optimalité de pareto 19

2.6 : Exemple de la relation de dominance. 21

2.7 :L'optimalité locale au sens de Pareto 22

2.8 Le théorème de contact[53] 23

2.9 Les niveaux de préférence dans la relation de dominance[53] 24

2.10 : Représentation de la surface de compromis 25

2.11 : Formes les plus courantes de la surface de compromis dans le cas de deux objectifs. 26

2.12 Représentation du point idéal et de point « nadir » 28

2.13 Propriété de convexité. 29

2.14 : La représentation de la surface de compromis. 29

2.15 :Classification des Méthodes d'optimisation multi-objectif. 33

3.1 .Evolution d'une solution dans la méthode de descente 49

3.2 .Fonctionnement de l'algorithme de recuit simulé. 52

3.3 Fonctionnement de l'algorithme génétique 57

3.4 Représentation chromosomique 58

3.5 Codage binaire de trois variables. 58

3.6 Croisement en un seul point. 60

3.7 Croisement en deux points. 60

3.8 Mutation d'un chromosome. 61

4.1 Deux fronts Pareto du problème de KUR 74

4.2 Deux fronts Pareto du problème de ZDT3. 75

4.3 .Nondominated solutions with NSGA-II on SCH 78

4.4 .Nondominated solutions with NSGA-II on KUR 79

4.5 .Nondominated solutions with NSGA-II on ZDT1 79

4.6 .Nondominated solutions with NSGA-II on ZDT2 80

4.7 .Nondominated solutions with NSGA-II on ZDT3 81

4.8 .Nondominated solutions with NSGA-II on ZDT4 82

4.9 .Nondominated solutions with NSGA-II on ZDT6 83

Introduction générale

La maximisation du profit est toujours la poursuite de l'humanité, quel que soit le domaine, en particulier l'évolution des enjeux énergétiques et des problèmes de population ces dernières années. Pour de nombreux problèmes sociaux et économiques, il est nécessaire de trouver une solution optimale pour réduire les intrants et augmenter la production. En conséquence, le problème de l'optimisation est de plus en plus préoccupé par les chercheurs. Ces dernières années, de nombreux chercheurs ont proposé de nombreuses méthodes d'optimisation de résolution de problèmes sous de nombreux aspects. Beaucoup d'entre eux consistent à trouver la valeur des variables de contrôle pour minimiser l'objectif unique. Cependant, de nombreux problèmes de décision dans le monde réel doivent non seulement prendre en compte un seul facteur, mais également une variété de facteurs pour obtenir une décision. En d'autres termes, de nombreux problèmes pratiques sont le problème de la programmation multi-objectifs. Par conséquent, ces dernières années, les chercheurs ont mené des recherches approfondies sur la planification multiobjective et obtenu de nombreux résultats de recherche significatifs, qui jettent les bases de la recherche future sur le problème multiobjectif. Ces dernières années, l'algorithme de recherche heuristique a connu un grand développement.

Certains chercheurs ont introduit et développé de nombreux algorithmes de recherche heuristique en fonction du comportement de l'animal social dans la nature et la caractéristique des problèmes d'optimisation à optimiser comme l'algorithme génétique [5], l'algorithme d'optimisation de l'essaim de particules [29], l'algorithme d'évolution différentielle [1], l'algorithme de saut de grenouille aléatoire [35], l'algorithme de colonie d'abeilles artificielles [12], etc. Dans tous ces algorithmes de recherche heuristique, l'algorithme génétique est très représentatif. Selon ses

bonnes performances de calcul et son idée génétique, l'algorithme génétique est largement utilisé dans de nombreux aspects [45, 54]. Le premier algorithme génétique est utilisé pour traiter le problème d'optimisation à objectif unique, tandis qu'au fil du temps, de nombreux chercheurs introduisent de nombreux algorithmes d'optimisation multiobjectifs selon l'idée de l'algorithme génétique. Ces algorithmes d'optimisation multiobjectifs contiennent un algorithme génétique d'évaluation vectorielle [41], un algorithme génétique multiobjectif [14, 36], un algorithme évolutionnaire de force Pareto [34] et un algorithme génétique de tri non dominé [28]. Après plusieurs années plus tard, afin de modifier chacun de ces algorithmes génétiques multiobjectifs calculant les performances, de nombreux chercheurs introduisent également la version améliorée de ces algorithmes génétiques multiobjectifs comme l'algorithme évolutionniste de force Pareto 2 [16] qui est la version améliorée de la force Pareto algorithme évolutif, force Algorithme évolutif de Pareto [37] qui est également la version améliorée de l'algorithme évolutionnaire de force Pareto, algorithme génétique de tri non dominé 2 qui est la version améliorée de l'algorithme génétique de tri non dominé[26] et ainsi de suite. De cette manière, de nombreux algorithmes génétiques multiobjectifs sont proposés au cours de ces années, qui ont largement contribué à ces problèmes d'optimisation multiobjectifs.

L'algorithme génétique de tri non dominé 2 (que nous appelons l'algorithme NSGA-II dans le reste de cet article) est la version améliorée de l'algorithme génétique de tri non dominé proposé pour la première fois par Deb en 2001. Au cours de ces années, le développement et l'utilisation de l'algorithme NSGA-II s'est avéré être un très bon algorithme génétique multiobjectif qui a été utilisé avec succès dans de nombreux problèmes d'optimisation multi-objectifs [4, 33] car les performances de calcul d'un algorithme varient beaucoup lorsqu'il s'agit de problèmes d'optimisation différents. En plus de cela, les performances de calcul de l'algorithme dépendent également de la stratégie d'optimisation à l'intérieur de l'algorithme lui-même. Ainsi, de nombreux chercheurs ont également proposé des stratégies d'amélioration dans l'algorithme NSGA-II pour remédier aux lacunes lors de l'optimisation de certains problèmes d'optimisation.

Dans ce mémoire, nous organiserons le contenu de la manière suivante :

•chapitre 01 :

Dans ce chapitre,nous présentons les principes de base de l'optimisation mono objectif(l'optimisation avec des variables dont le domaine est contenue et l'optimisation combinatoire dont le domaine est discret).nous donnerons aussi des définitions et principaux concepts d'un problème optimisa-

tion Mono-objectif et une classification générale des méthodes d'optimisation mono-objectif.Puis on présentera quelques une de ces méthodes pour la résolution d'un problème d'optimisation mono-objectif.

•chapitre 02 :

Dans ce chapitre, nous présentons l'optimisation multi-objectif qui sera le cadre de travail de cette thèse. Nous introduisons les concepts fondamentaux tels que la dominance, la surface de compromis. Nous décrivons aussi les principales approches non Pareto de résolution pour ces problèmes.

•chapitre 03 :

Dans ce chapitre, nous présentons quelques méthodes de résolution Pareto fondées sur des métaheuristiques. Ainsi, nous introduisons les méthodes de recherche locale et les approches évolutionnaires. Nous décrivons aussi deux algorithmes évolutionnaires représentatifs adoptant une approche Pareto pour résoudre les problèmes d'optimisation multi-objectifs.

•chapitre 04 :

Ce chapitre a présenté une nouvelle stratégie d'amélioration de l'algorithme NSGA II, qui utilise une méthode de recherche locale pour rechercher une meilleure solution autour de la solution optimisée de l'algorithme NSGA-II.

Chapitre 1

OPTIMISATION MONO-OBJECTIF

1.1 Définition d'un problème d'optimisation

Un problème d'optimisation est défini par un espace d'état, une ou plusieurs fonctions objectives et un ensemble de contraintes.

- Espace d'état

est défini par l'ensemble des domaines de définition des variables du problème.

- Les variables

Du problème peuvent être de nature diverse (réelle, entière,bivalent, etc.) et sont exprimées de données qualitatives ou quantitatives.

- Fonction objective

Représente le but à atteindre pour le décideur(minimisation de coût, de durée, d'erreur, ...). Elle définit un espace de solutions potentielles au problème.

- L'ensemble de contraintes

Sur l'espace d'état que les variables doivent satisfaire. Ces contraintes sont souvent des contraintes d'inégalités ou d'égalités et permettent en général de limiter l'espace de recherche.

Le problème d'optimisation (P) peuvent être formulés comme suit :

$$(P): \begin{cases} optimiser f(x) \\ x \in \Omega \end{cases} \tag{1.1}$$

avec $f : \mathbb{R}^n \to \mathbb{R}$ est une fonction objectif ou bien la fonction optimiser sur l'ensemble Ω

1) Si $\Omega = \mathbb{R}^n$ alors le problème est dit problème d'optimisation sans contrainte
2) Si $\Omega = \{x \in \mathbb{R}^n / g_i(x) \leq 0 \quad \forall i \in 1, ..., m, h_j(x) = 0 \quad \forall j \in 1, ..., m'\}$ et $g_i, h_i : \mathbb{R}^n \to \mathbb{R}$ des fonctions de contraintes, alors le problème est dit problème d'optimisation avec contraintes mixtes.
3) Le problème est dit problème d'optimisation avec contraintes égalités : si $\Omega = \{x \in \mathbb{R}^n, h_i(x) = 0\}$
4) Si $\Omega = \{x \in \mathbb{R}^n, g_i(x) \leq 0\}$ alors :
   P est dit problème d'optimisation avec contraintes d'inégalités.
5) L'ensemble des points $\{x \in \mathbb{R}^n / g_i(x) \leq 0 \quad \forall i \in 1, ..., m, h_j(x) = 0 \quad , \forall j \in 1, ..., m'\}$:est appelé ensemble des candidats.
   On appelle solution du problème d'optimisation tout minimum/maximum global de la fonction f restreinte à cet ensemble des candidats noté x^* .
6) Les fonctions f,g et h sont linéaires,alors (P) est dit problème d'optimisation linéaire
7) Si f,g et h sont convexe,alors (P) est dit problème d'optimisation convexe.
8) Si le problème

$$(P): \begin{cases} optimiser f(x) = \frac{1}{2} X^t A X \\ AX \leq 0 \end{cases} \tag{1.2}$$

   Est appelé problème d'optimisation quadratique.

1.1.1 Terminologie de la solution

• **Zone de solution**

Ensemble de toutes les contraintes du modèle sont satisfaite.

• **Solution réalisable(admissible)**

Solution pour laquelle toutes les contraintes sont satisfaites(appartenant au domaine réalisable).

• **Solution optimale**

Solution ayant la meilleure valeur possible de l'objectif.

1.1.2 Minimum et Maximum local

Soit $f : \mathbb{R}^n \to \mathbb{R}$

1. Minimum local : un point x^* est un minimum local de la fonction f s'il existe $\epsilon > 0$ tel que

$$f(x^*) \leq f(x) \text{ pour tout } x \in v_\epsilon(x^*)$$

avec $v_\epsilon(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| < \epsilon\}$.

2. Maximum local :un point x^* est un maximum local de la fonction f si

$$f(x^*) \geq f(x) \text{ pour tout } x \in v_\epsilon(x^*)$$

1.1.3 Minimum et Maximum global :

1. Minimum global : un point x^* est un minimum global de la fonction f si

$$f(x^*) \leq f(x) \text{ pour tout } x \in \mathbb{R}^n$$

2. Maximum global : un point x^* est un maximum global de la fonction f si

$$f(x^*) \geq f(x) \text{ pour tout } x \in \mathbb{R}^n$$

1.1.4 Notion de convexité :

Définition :

Soit $D \subset \mathbb{R}^n$.On dit que
D est convexe $\lambda x + (1-\lambda)y \in D \quad \forall (x,y) \in D$ et $\forall \lambda \in [0,1]$

$$\Leftrightarrow \sum_{i=1}^{k} \lambda_i x_i \in X \quad \forall (x_i)_{i=1\ldots k} \in D \text{ et } \quad \forall \lambda_i \geq 0 \text{ t.q } \sum_{i=1}^{k} \lambda_i = 1$$

Définition d'un fonction convexe :

Soit l'ensemble convexe $D \subset \mathbb{R}^n$. Une fonction $f : D \to \mathbb{R}$ est convexe si et seulement si $\forall x, y \in D, \forall \lambda \in [0,1]$,alors $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$. **Remarque :** En changement le sens de l'inégalité de la définition d'une fonction convexe on obtient le définition d'une fonction concave.

1.1.5 Gradient et Matrice Hessien :

Définition :

(Le gradient et la matrice hessienne) : Le gradient d'une fonction $f \in \mathbb{R}^n \to \mathbb{R}$ continûment différentiable évalué au point $x \in \mathbb{R}^n$ s'écrit :

$$\nabla f(x^*) = (\frac{\partial f(x)}{\partial x_1} \frac{\partial f(x)}{\partial x_2} \ldots\ldots \frac{\partial f(x)}{\partial x_n})^T$$

et l'élément de la i^{eme} ligne j^{eme} colonne de la matrice Hessienne s'écrit :

$$[\nabla^2 f(x)]_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$$

1.2 Classification générale des méthodes d'optimisation mono-objectif

La résolution d'un problème est réalisée à l'aide des méthodes d'optimisation dont la classification est illustrée dans la figure 1.1.

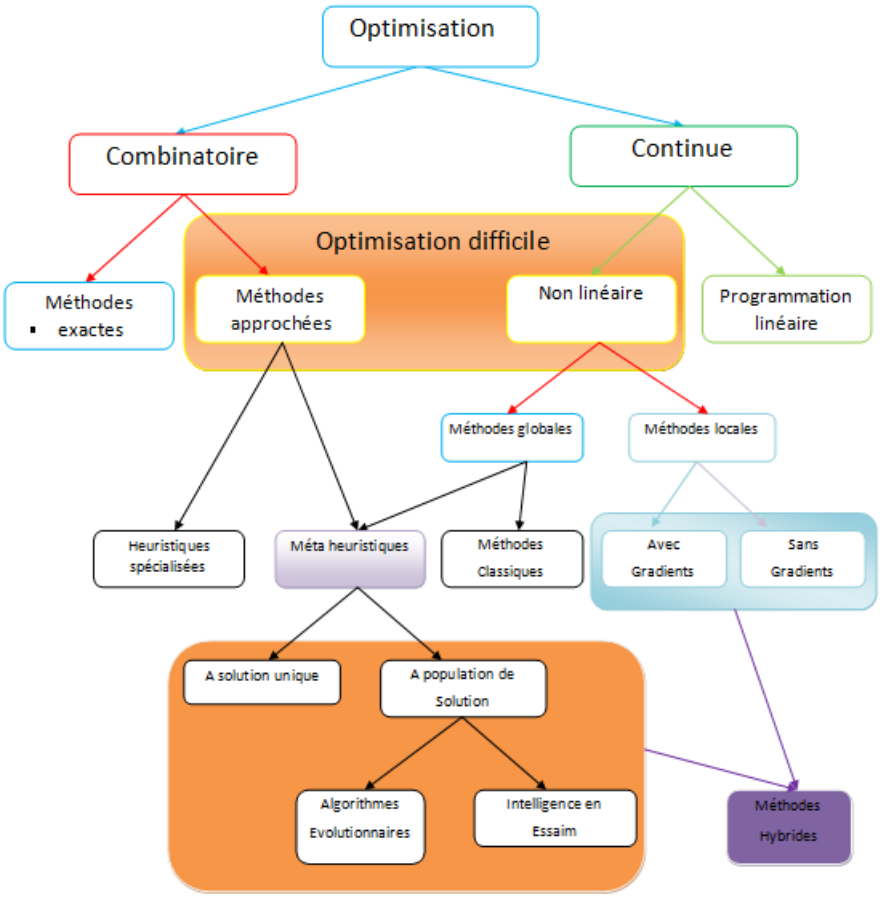


FIGURE 1.1 – Classification générale des méthodes d'optimisation mono-objectif [53]

Complexité des problèmes et efficacité des algorithmes :

La théorie de la complexité s'attache à classifier les problèmes selon leur difficulté.

Un problème de décision : c'est un problème dont la réponse est tout simplement OUI ou NON.

▪ P : la classe des problèmes qui peuvent être résolus en temps polynomial sur une machine déterministe.

Ces problèmes sont souvent considérés comme ceux pour lesquels il existe un algorithme efficace.

▪ NP : la classe des problèmes qui peuvent être résolus en temps polynomial sur une machine non déterministe.

Problèmes faciles et difficiles

▪ Un problème est facile s'il appartient à la classe P pour lesquels il existe un algorithme efficace pour les résoudre.

▪ Un problème de décision est difficile s'il appartient à la classe NP-Complet,lesquels il n'existe pas à l'heure actuelle un algorithme efficace pour les résoudre,Cependant,il existe un moyen pour vérifier ,en temps polynomial ,la véracité d'une solution donné.

▪ Un problème d'optimisation appartient à la classe NP-Difficile si son problème décisionnel appartient à la classe NP-Complet.

Chapitre 2

OPTIMISATION MULTI-OBJECTIF

2.1 Structure d'un problème d'optimisation multi-objectifs

Un problème multi-objectif ou multi-critère peut être défini comme un problème dont on recherche l'action qui satisfait un ensemble de contraintes et optimise un vecteur de fonctions objectifs. Les problèmes d'optimisation ont en général plusieurs solutions car la définition d'un optimum ne peut pas être établie dans les problèmes multi objectifs [3]
Un problème d'optimisation multi-objectifs (PMO)(multi-objective optimisation problem)est posé sous la forme suivante :

$$\begin{cases} "optimiser" F(x) = (f_1(x), f_2(x), .., f_k(x)) \\ x \in S \end{cases}$$

où :

• K : Le nombre de fonctions objectifs,telle que $K \geq 2$

• $x = (x_1, x_2, ..., x_n)^{'}$: Le vecteur représentant les variables de décision .

• $S = \{x \in \mathbb{R}^n : g_i(x) \leq 0 \quad i = 1, 2, .., m$ et $x \geq 0\}$ est l'ensemble des solutions admissibles (ou réalisables) du problème.

• $f_i(x)$: fonction objectif à minimiser, tels que :

$$f_i(x) : S \to \mathbb{R} \quad \forall \quad i = 1, .., k$$

• $g_i(x)$: fonction représentant la contraintes du problème, où :

$$g_i(x) : \mathbb{R}^n \rightarrow \quad i = 1, .., m$$

L'image de l'ensemble des solutions admissibles S, appelée ensemble des critères, est donnée par :

$$Z = f(s) = \{z \in \mathbb{R}^k : z = f(x), \forall x \in S\},$$

où :$Z = (z_1, z_2, ..., z_k)$ est le vecteur des critères, tel que $z_i = f_i(x)$.

Définition et Vocabulaire

Dans ce qui suit, quelques définitions nécessaires à la compréhension de certains concepts de base que nous utiliserons dans la suite.
On considère un problème multi-objectif ayant n paramètres (variables de décisions) et k objectifs à optimiser (critères) [53] :

Vecteur de décision :

Une action(ou un vecteur de décisions) sera notée :

$$x = (x_1, x_2, \ldots, x_n)$$

avec x_i les variables du problème et n le nombre de variables c'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction f.[3]

L'ensemble de contraintes

Les contraintes seront notées :

$$g_i(x) \quad \text{avec} \quad i = 1, ..., m$$

et m le nombre de contraintes.

Fonction objectif :

Le vecteur de fonctions objectifs sera notéf :

$$f(x) = (f_1(x), f_2(x), \ldots, f_K(x))$$

avec f_i les objectifs (ou critères de décision) et k le nombre d'objectifs,la fonction que l'algorithme d'optimisation va devoir "optimiser" (trouver un optimum)

Espace de recherche

Espace de recherche : représentant l'ensemble des valeurs pouvant être prises par les variables.

Espace réalisable

Espace réalisable : représentant l'ensemble des valeurs des variables satisfaisant les contraintes.

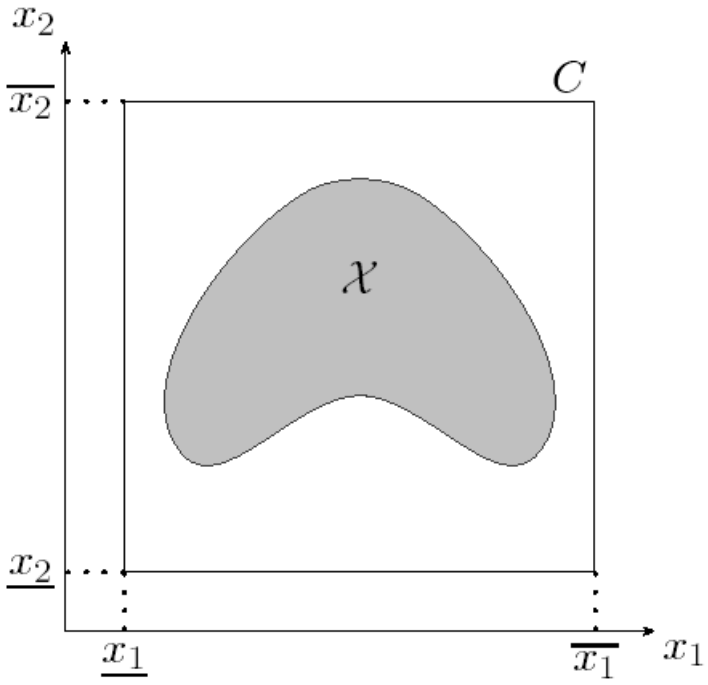


FIGURE 2.1 – Espace de recherche et espace réalisable

χ : l'espace réalisable

C : l'espace de recherche

Un problème d'optimisation recherche l'action x^* telle que les contraintes $g(x^*)$ soient satisfaits pour $i = 1, .., m$ et qui optimiser la fonction f :$f(x^*) = (f_1(x), f_2(x), \ldots, f_K(x))$

L'union des domaines de définition de chaque variable et les contraintes forment un ensemble E que nous appelons l'ensemble des actions réalisables.

Nous appellerons F l'ensemble des objectifs réalisables.

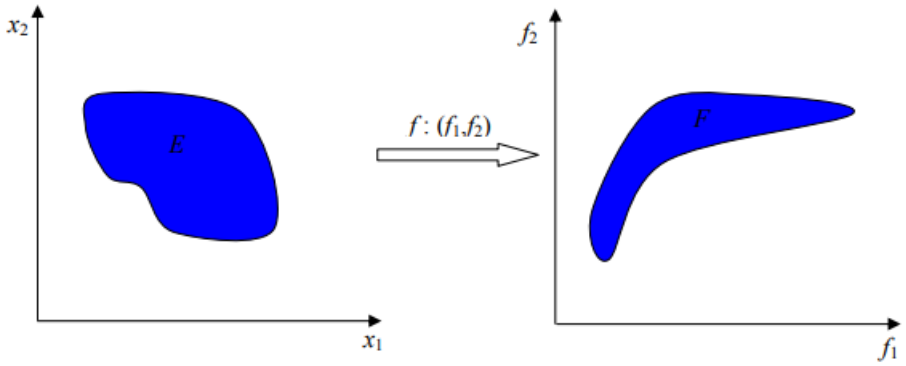


FIGURE 2.2 – définition de E,F et f.

Après avoir trouvé les solutions du problème multi-objectif que d'autres difficultés surviennent : il faut sélectionner une solution dans cet ensemble. La solution qui sera choisie par l'utilisateur va refléter les compromis opérés par le décideur vis-à-vis des différentes fonctions objectifs [6]. En résumant ces notions mathématiques représentant un problème d'optimisation multi-objectif par la figure suivante :

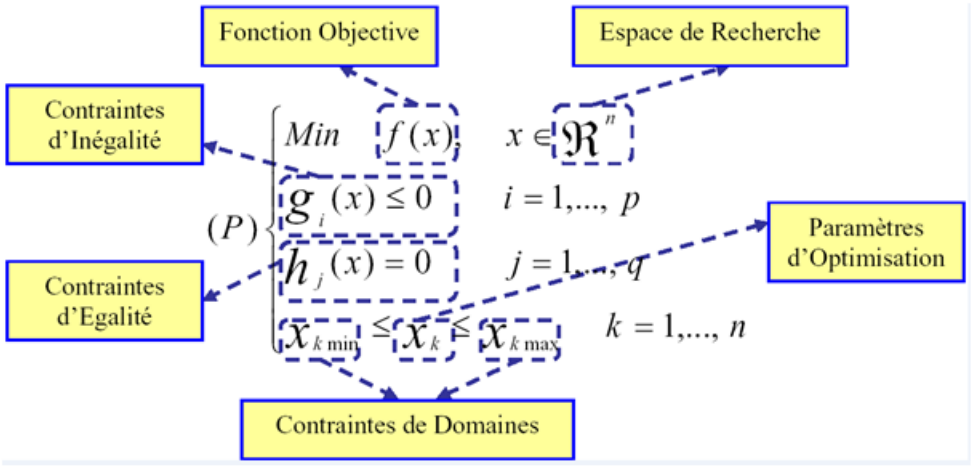


FIGURE 2.3 – La présentation mathématique d'un problème d'optimisation multi-objectif. [6].

Remarque

Dans le cas de fonction f de maximisation, il suffit de minimiser f [6].

2.2 Les variables d'un problème multi-objectif

Les variables du problème sont les paramètres dont le concepteur doit ajustés pour modifier le système à concevoir. Il y a plusieurs types de ces variables :

Variables indépendantes

Sont les quantités actuelles dont le concepteur s'occupe directement, comme la géométrie, les propriétés du matériel, le volume de production, la surface finale, la configuration des composants, les propriétés de lubrification et autres .... Les variables indépendantes sont usuellement connues comme les variables de problème ou les paramètres de problème.

Variables dépendantes

Sont les variables dont le concepteur ne peut attribuer des valeurs mais il les utilise pendant les paramètres du problème. Les variables dépendantes sont usuellement nommées les caractéristiques ou les attributs du problème. Exemple de caractéristiques du problème l'énergie de consomption, le prix et l'erreur de contrôle. La valeur du problème est largement une fonction des caractéristiques du problème. Dans l'optimisation, la valeur de la fonction objective correspond à la valeur du caractéristique particulière. Une fonction objective c'est ainsi la relation entre les paramètres du problème et les valeurs des caractéristiques particulières. Pour un problème d'optimisation général, il faut être très difficile ou bien impossible de représenter analytiquement tous comme des relations. Généralement, les caractéristiques doivent être le résultat de la simulation complexe, ou ils doivent inclure en quantifiable le jugement de l'homme.

Variables d'état

Sont le type intermédiaire des variables de problème entre les variables dépendantes et les variables indépendantes, comme la pressure de cylindre hydraulique ou le courant dans un moteur

électrique. On ne peut pas attribué directement des valeurs aux variables d'état, aussi ces variables ne peuvent pas directement contribuer dans la valeur de problème comme les caractéristiques.

Variables opérationnelles

Sont les variables qui peuvent être changées par l'opérateur après la conception actuellement construise.

Variables environnement

Où les variables externes sont des facteurs écologiques qui affectent le problème au cours de l'utilisation, comme le changement de charge, la température extrême et celle d'usage. Le concepteur a pour déterminer les conditions de travail du problème les variables de l'environnement et les variables opérationnelles dans ses expériences.
Le problème peut être formulé quant à l'attribution des valeurs aux paramètres du problème pour être sûr que les variables d'état et les caractéristiques sont les meilleures au cours du champ large des variables opérationnelles et de l'environnement. Ça nécessite une optimisation de problème multi-objectifs complexe [2].

2.3 Multiplicité des solutions

Lorsque l'on cherche à obtenir une solution optimale à un problème d'optimisation multi-objectif donné, on s'attend souvent à trouver une solution et une seule. En fait, on rencontre rarement ce cas de figure. La plupart du temps, on trouve une multitude de solutions, du fait que certains des objectifs sont contradictoires[22, 53].

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 [3].

2.4 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 en prenant en compte l'ensemble des critères.
La différence entre ces deux communautés s'exprime dans le schéma ci dessous :

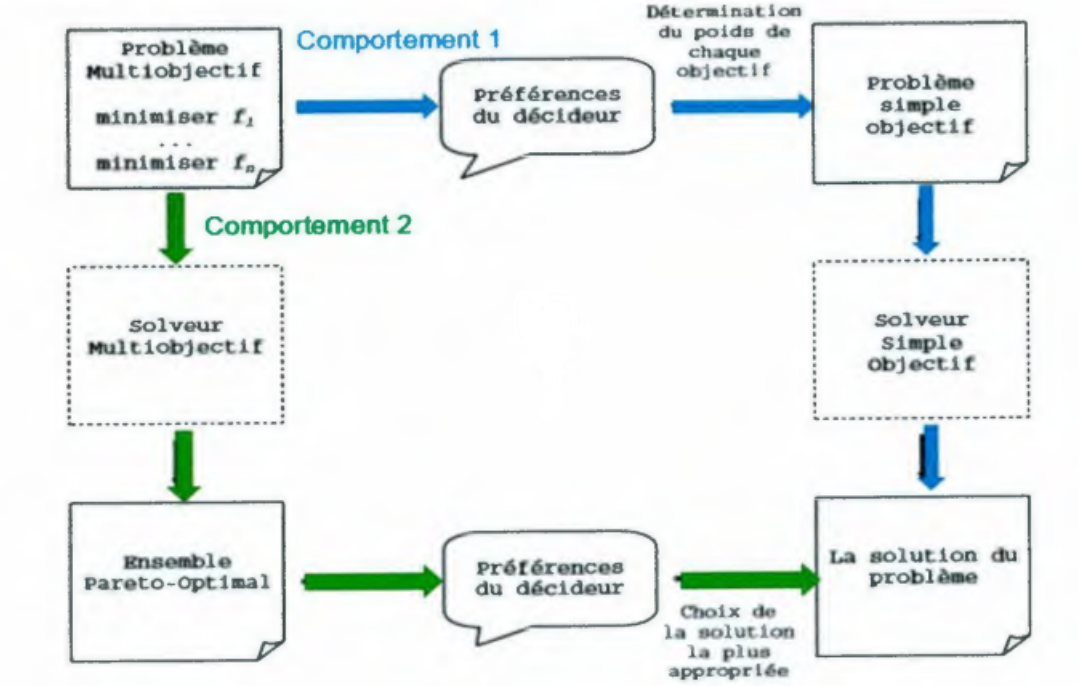


FIGURE 2.4 – Modes de résolution d'un problème Multi-objectif

[3, 6].

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) [6, 19].

2.5 Classification des problèmes multi-objectifs :

On peut classer les différents problèmes d'optimisation que l'on rencontre dans la vie courante en fonction de leurs caractéristiques :

1- Nombre de variables de décision :

une ⇒ mono variable.

Plusieurs ⇒ multi variable.

2- Type de variable de décision :

Nombre réel continu ⇒continu.

Nombre entier ⇒ entier ou discret.

Permutation sur un ensemble fini de nombres ⇒ combinatoire.

3- Type de fonction objective :

Fonction linéaire des variables de décision ⇒ linéaire.

Fonction quadratique des variables de décision ⇒ quadratique.

Fonction non linéaire des variables de décision ⇒ non linéaire.

4- Formulation de problème :

Avec contraintes ⇒ contraint.

Sans contraintes ⇒ non contraint [53].

2.5.1 Approches Pareto

Les approches Pareto ne transforment pas les objectifs du problème, ceux-ci sont traités sans aucune distinction pendant la résolution. Dans les paragraphes suivants, nous définissons tout d'abord la notion de dominance au sens de Pareto, la frontière de Pareto, Pareto optimal. Ensuite, nous présentons les techniques évolutionnaires utilisant cette notion.

La dominance au sens de Pareto

L'idée d'utiliser la dominance au sens de Pareto a été proposée par Goldberg pour résoudre les problèmes multi-objectifs. Il suggère 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. Ce concept ne permet pas de choisir une alternative plutôt qu'une autre mais il apporte une aide précieuse au décideur.

Vilfredo Pareto est un mathématicien italien (Pareto, 1896). Il a posé les bases de la solution d'un

problème économique multi-objectif : « Dans un problème multi-objectif, il existe un équilibre tel que l'on ne peut améliorer un critère sans détériorer au moins un des autres ». Cet équilibre est appelé optimum Pareto. Donc une solution x est dite Pareto optimale si elle n'est dominée par aucune autre solution appartenant à l'espace réalisable X. Ces solutions sont appelées solutions non dominées, ou non inférieures.

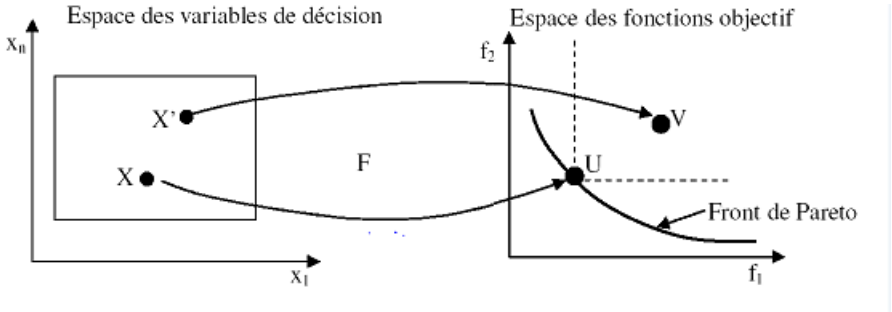


FIGURE 2.5 – Dominance de pareto ou optimalité de pareto

La Dominance

On dit que la solution $x = (x_1, x_2, .., x_n)'$ domine la solution $y = (y_1, y_2, .., y_n)$ si :

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

Mathématiquement , on dit qu'une solution x domine une solution y (dans le cas de minimisation) ssi :

$$\forall\ i \in \{1, ..., k\}, f_i(x) \leq f_i(y) \text{ et } \exists\ i \in \{1, ..., k\} \text{ tel que } f_i(x) < f_i(y) \text{ et note } x \leq y$$

Dominance forte

On dit que la solution $x = (x_1, x_2, ..., x_n)^{'}$ domine fortement la solution $y = (y_1, y_2, .., y_n)^{'}$ si et seulement si x est meilleure que y sur tous les critères.

Mathématiquement, on dit qu'une solution x domine fortement une solution y (dans le cas de minimisation) ssi :

$$\forall i \in \{1, .., k\}, f^i(x) < f_i(y) \text{ et note } x < y$$

La dominance faible

On dit que la solution $x = (x_1, x_2, ..., x_n)^{'}$ domine faiblement la solution $y = (y_1, y_2, .., y_n)^{'}$ si et seulement si

$$i \in \{1, ..., k\} \quad f_i(x) \leq f_i(y)$$

Remarque

Notons que pour toute paire de solution y et z un et un seul des cas suivants peut se présenter :

– y domine z

– y est domine par z

– y et z sont équivalentes au sens de la dominance.

Les solutions équivalentes au sens de la dominance sont appelées dans se qui suit,solutions équivalentes au sens de Pareto ou solutions Pareto équivalentes ou encore solutions non dominées .

Propriétés de la relation de dominance

Notation :

Si le vecteur X domine le vecteur Y, on note alors $X < Y$ la relation binaire de dominance.

Définition :

La relation binaire de dominance notée :

▪ Elle n'est pas réflexive car une solution ne se domine pas elle même.

▪ Elle n'est pas symétrique, car on n'a jamais $X < Y$ et $Y < X$

▪ Elle est transitive, car si $X < Y$ et $Y < Z$ alors $X < Z$.

▪ Elle n'est pas antisymétrique, en raison de l'existence de solutions Pareto-équivalentes.

Au final, on conclut que la relation de dominance est donc une relation d'ordre partiel strict sur l'espace des critères réalisables [52].

Remarque

Notons que $X < Y$ n'implique pas $Y < X$

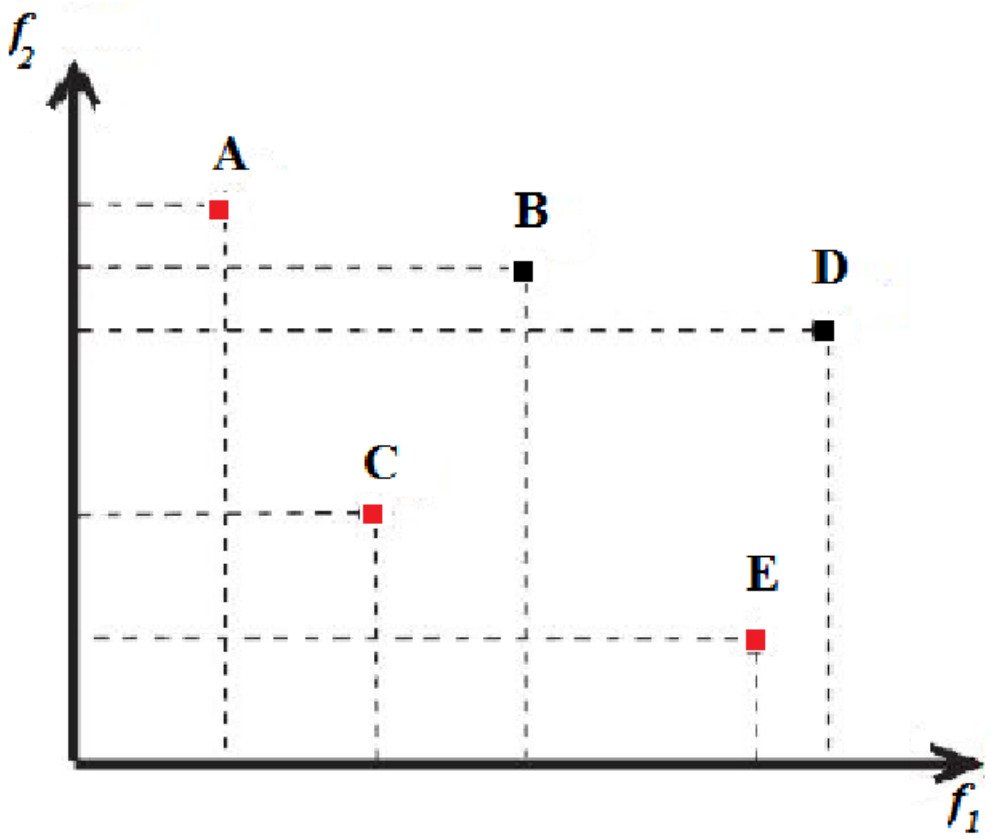


FIGURE 2.6 – : Exemple de la relation de dominance.

Dans cette exemple, les points A, C, E ne sont pas dominés par d'autres points. Tandis que le point C domine les points B et D, le point E domine le point D. Les points A, C, E sont au même niveau, c'est-à-dire, ils sont incomparables.

2.5.2 Optimalité de pareto

Soit $[x_1, x_2, .., x_n]^T$ un vecteur de décision avec : $x \in X$ (L'espace réalisable) x est dit Pareto optimal, s'il n'existe pas une solution y domine x. Une solution Pareto optimal appelée aussi : solution efficace, non inférieur ou non dominée solution.

Optimalité locale au sens de Pareto

Un vecteur $\vec{x} \in \mathbb{R}^n$ est optimal localement au sens de Pareto s'il existe un réel $\delta > 0$ tel qu'il n'y ait pas de vecteur $\vec{x}'$ qui domine le vecteur $\vec{x}$ avec $\vec{x}' \in \mathbb{R}^n \cap B(\vec{x}, \delta)$ où $B(\vec{x}, \delta)$ représente une boule de centre $\vec{x}$ et de rayon δ.

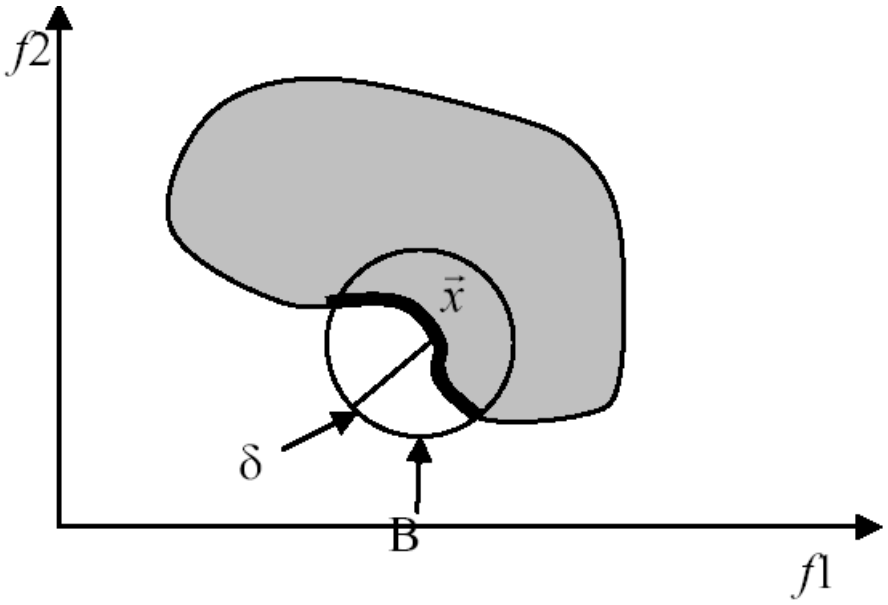


FIGURE 2.7 – :L'optimalité locale au sens de Pareto

Optimalité globale au sens de Pareto

Un vecteur $\vec{x}$ est optimal globalement au sens de Pareto (ou optimal au sens de Pareto) s'il n'existe pas de vecteur $\vec{x}'$ tel que $\vec{x}'$ domine le vecteur $\vec{x}$

Théorème de contact

On dit aussi qu'un vecteur $\vec{x}$ est optimal au sens de $Pareto$ pour un problème d'optimisation multi-objectifs donné si :

$(\vec{C} + \vec{x}) \cap F = \{\vec{x}\} \Rightarrow$ Le théorème de contact (figure 2.8).

Où F désigne l'espace des solutions réalisable.

Cône négatif :

Un cône négatif est définit dans $\mathbb{R}^k$ de la manière suivante :

$$c^- = \{\vec{x}\ \vec{f}(\vec{x}) \in \mathbb{R}^k \text{ et } \vec{f}(\vec{x}) \leq 0\}$$

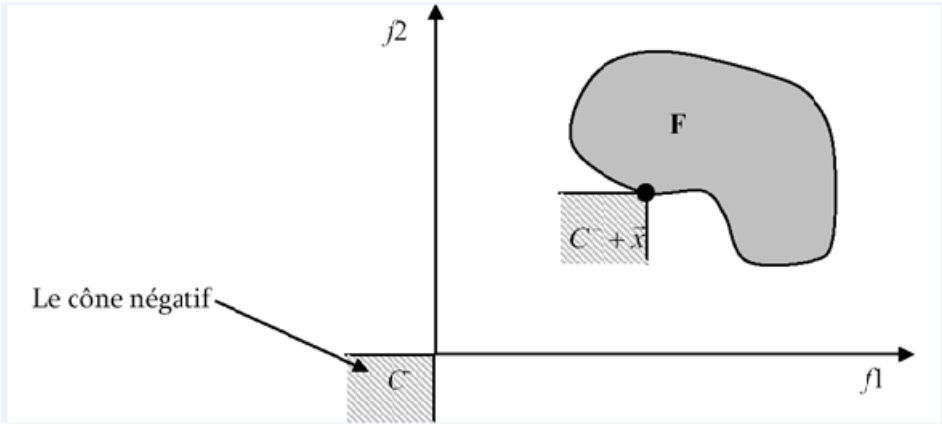


FIGURE 2.8 – Le théorème de contact[53]

.

Les niveaux de préférence dans la relation de dominance

Lorsqu'on applique la définition de la dominance, on peut définir quatre régions auxquelles on peut attribuer des niveaux de préférence. Ces régions sont présentées à la figure 05. Cette figure reprend le découpage défini par le cône négatif que l'on a introduit précédemment et l'étend à tout l'espace.

Par exemple, si ce graphique est centré sur une solution A et que l'on compare cette solution avec une solution B, on aura les possibilités suivantes :

- Si la solution B se trouve dans le quadrant 1, alors la solution A est préférée à la solution B.
- Si la solution B se trouve dans le quadrant 3, alors la solution A est dominée par la solution B.
-Si la solution B se trouve dans l'un des quadrants 2 ou 4, alors on ne peut pas prononcer sur la préférence de A par rapport à B ou B par rapport à A.

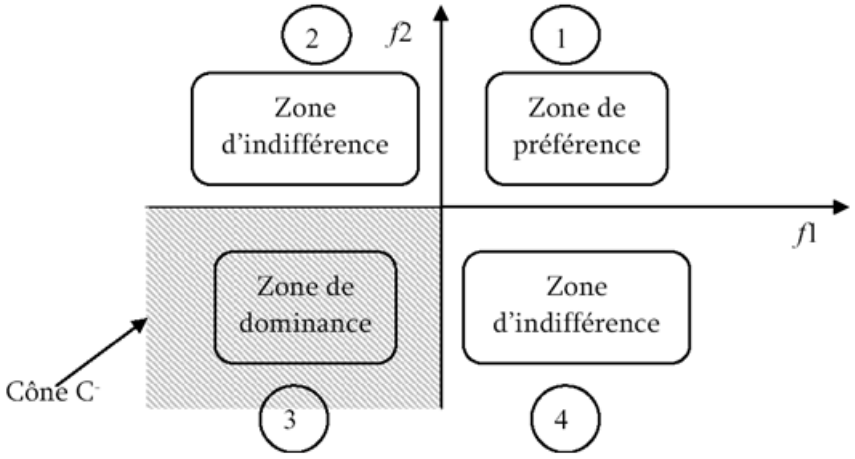


FIGURE 2.9 – Les niveaux de préférence dans la relation de dominance[53]

.

2.6 Surface de compromis

Les solutions obtenues lors du classement basé sur le principe de dominance de Pareto forme ce que l'on appelle la surface de compromis (ou Pareto Front).

Imaginons que nous avons un problème à deux objectifs (minimiser $f1$ et minimiser $f2$ sous les contraintes $\vec{f}(\vec{x}) \leq 0$ et $\vec{h}(\vec{x}) = 0$) :

- On appelle S l'ensemble de valeurs du couple $(f1(\vec{x}), f2(\vec{x}))$ quand à $\vec{x}$ respecte les contraintes $\vec{g}(\vec{x})$ et $\vec{h}(\vec{x})$

- On appelle P la surface de compromis.

On représente S et P sur la figure suivante :

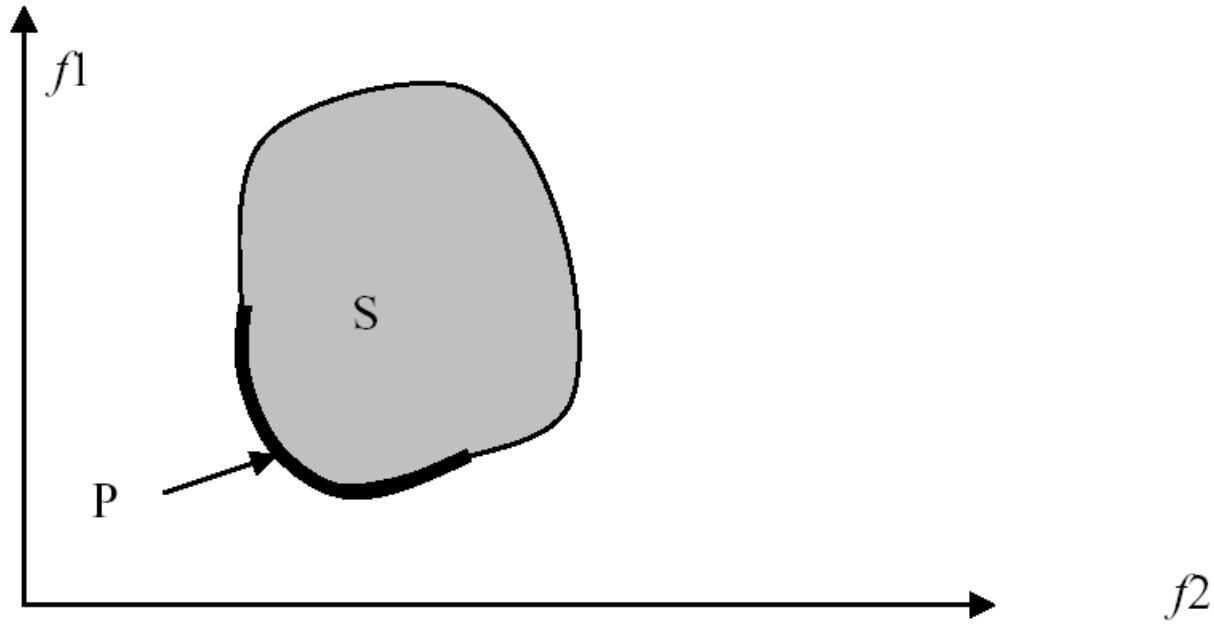


FIGURE 2.10 – : Représentation de la surface de compromis [53].

Une propriété est remarquable : en fonction du type du problème que l'on cherche à résoudre, on obtient une forme de surface de compromis. Les formes les plus courantes de surfaces de compromis sont réunies à la figure 2.11. Ces formes de surface de compromis sont typiques pour un problème d'optimisation multi-objectifs sur un ensemble de solutions convexes. C'est se type d'ensemble que l'on rencontre la plupart de temps.

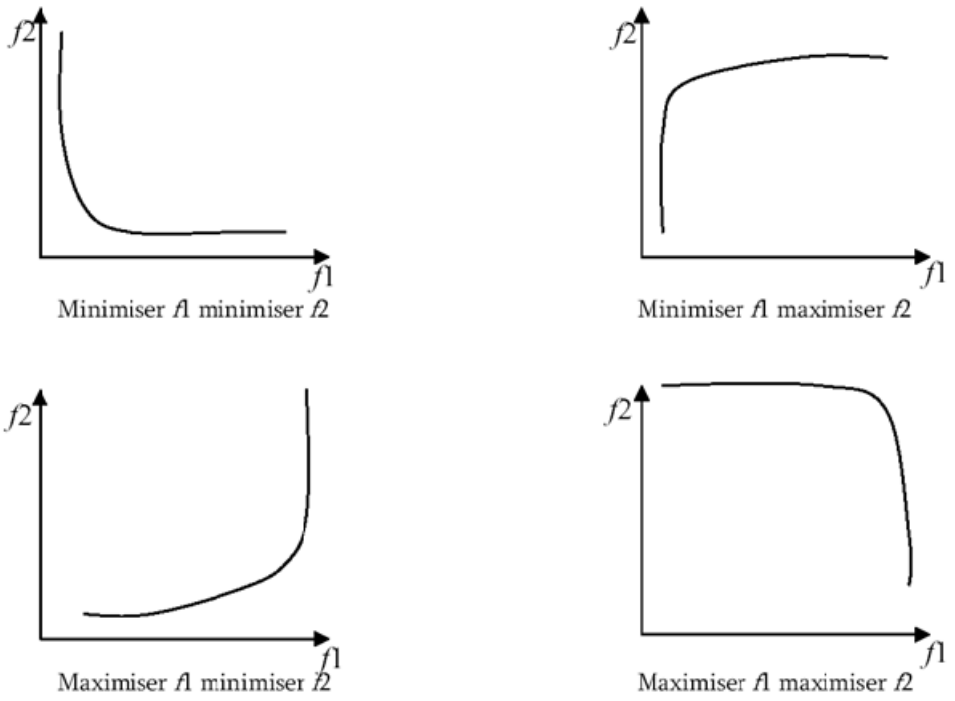


FIGURE 2.11 – : Formes les plus courantes de la surface de compromis dans le cas de deux objectifs.

On observe deux points caractéristiques associés à une surface de compromis :

Point idéal

Le point idéal (pi) est un vecteur de $\mathbb{R}^k$ dont les coordonnées sont les valeurs minimales de chacune des fonctions objectifs pris séparément. D'une autre manière, c'est le vecteur qui minimise chacune des fonctions objectifs f_i pris séparément.

$$Z^* = (z_1^*, z_2^*, ..., z_n^*) \text{ avec } z_i^* = min(f_i(x)) \quad x \in s \text{ et } \forall i = 1, .., n$$

Remarque

— Le point idéal n'appartient pas à l'espace des critères réalisables car les objectifs sont contradictoires.

— Le point idéal est appelé aussi vecteur des valeurs les plus préférées de chaque objectif f_i , ainsi le point idéal peut être utilisé comme point de référence.

— Le point de référence est un vecteur qui définit les valeurs souhaitables (buts) qu'il faut atteindre par chaque objectif f_i.

Point nadir

le vecteur nadir (z^{nad}) correspond aux bornes supérieures de chaque objectifs sur la surface de Pareto et non pas dans tout l'espace réalisable S.

$$z^{nad} = (z_1^{nad}, z_2^{nad}, .., z_k(nad) \text{ avec } z_i^{nad} = max(f_i(x))\ x \in X \quad pareto \quad \forall i = 1, .., k$$

Remarque

On peut définir le point nadir de la matrice des gains, où (Z^{nad}) se trouve sur le maximum de la ligne de la matrice des gains pour un problème à minimiser.

— Le point nadir est le vecteur des valeurs les plus mauvaises de chaque critères f_i ,il est utilisé comme point de réservation.

— Le point nadir sert à restreindre l'espace de recherche, il est utilisé dans certaines méthodes d'optimisation interactives.

— A titre d'exemple, on peut utiliser le point idéal et le point nadir pour la normalisation des valeurs des objectifs par :

$$f_i^{norm} = f_i - z_i / z_i^{nad} - z_i$$

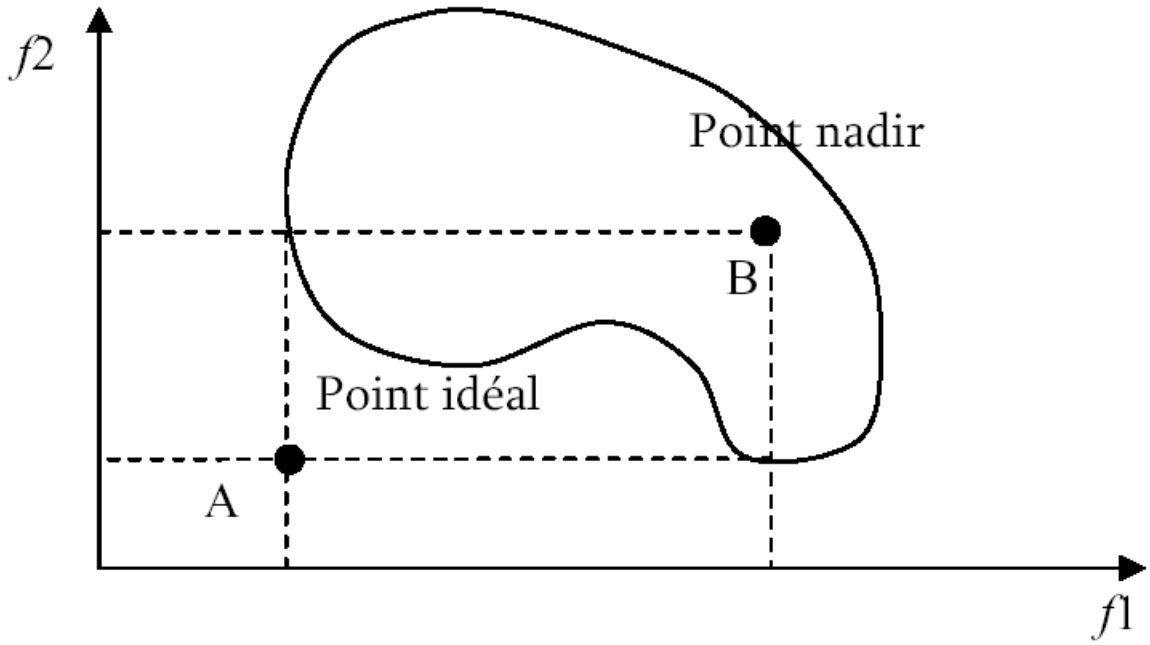


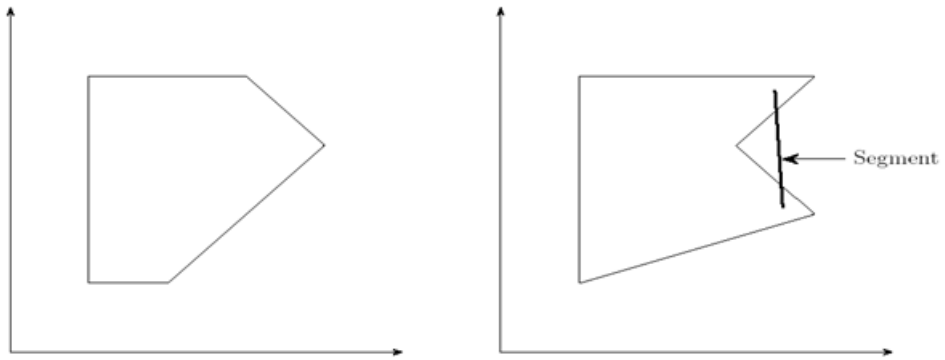
FIGURE 2.12 – Représentation du point idéal et de point « nadir »
[53].

Convexité

Un ensemble A est convexe, si et seulement si l'équivalence suivante est vérifiée :

$$x \in A \Leftrightarrow segment(x, y) \subset A$$

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).

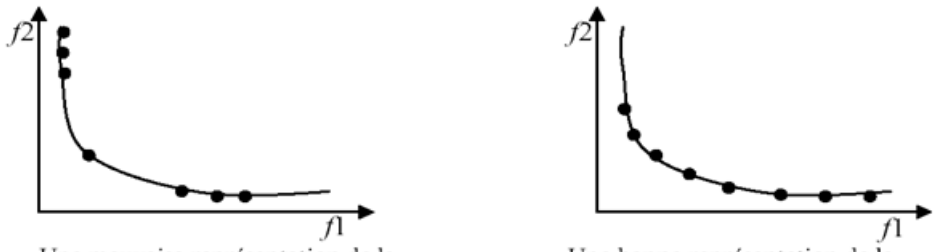


(a) Ensemble convexe. (b) Ensemble non convexe.

FIGURE 2.13 – Propriété de convexité.

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(figure 2.14 : la représentation de la surface de compromis).



Une mauvaise représentation de la surface de compromis.

Une bonne représentation de la surface de compromis.

FIGURE 2.14 – : La représentation de la surface de compromis.

Dans le premier cas, 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 utiles. En effet, s'il décide que la solution qu'il avait choisie ne lui convient pas, le choix d'une autre solution risque de se faire varier brusquement tous ses objectifs, et cette nouvelle solution ne lui conviendra pas non plus. Il est alors probable que la solution offrant le « meilleur » compromis se trouve dans une zone qui ne soit pas représentée par des points solution.

2.7 Efficacité (Ou Pareto optimale) :

Définition

Une solution $x^* \in X$ est une solution efficace s'il n'existe pas de $x \in X$ tel que $z(x)$ domine $z(x^*)$ [42].

Efficacité faible :

Une solution $x^* \in X$ est une solution faiblement efficace s'il n'existe pas de $x \in X$ telle que $z(x) < z(x^*)$.
Autrement dit, une solution est fortement efficace si son vecteur objectif est fortement dominée.

Efficacité forte :

Une solution $x^* \in X$ est une solution fortement efficace s'il n'existe pas de$x \in X$ telle que $x \neq x^*$ et $z(x) = z(x^*)$.
Autrement dit, une solution est faiblement efficace si son vecteur objectif est faiblement dominée.
Remarque
Toute solution efficace est faiblement efficace et l'inversent est faux.
Remarque
La notion de dominance se réfère à l'espace des critères alors que la notion d'efficacité se réfère à l'espace de décision.

2.7.1 Conditions nécessaires d'optimalités

Comme en optimisation mono-objectif un point solution d'un problème d'optimisation multi-objectif sous contraintes vérifie des conditions du premier ordre (c'est -à-dire portant sur la dérivée de la fonction coût et des contraintes) qui donnent lieu à un système d'équation vérifié par le point solution et des paramètres associés aux contraintes , les coefficients de Lagrange

Théorème

Soient $f : \mathbb{R}^n \to \mathbb{R}^k$ $, g : \mathbb{R}^n \to \mathbb{R}^P$ $, h : \mathbb{R}^n \to \mathbb{R}^p$ des fonctions de classe C^1.si x^* est solution du problème d'optimisation muiti-objectif ,Alors il existe $\omega \in \mathbb{R}^k, \mu \in \mathbb{R}^n, \lambda \in \mathbb{R}^p$ Avec $(\omega, \mu, \lambda) \neq (0,0,0) \subset \mathbb{R}^k_+ * \mathbb{R}^m_+ * \mathbb{R}^P$ tels que :

$$\begin{cases} \sum_{i=1}^{k} \omega_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) + \sum_{i=1}^{k} \lambda_l \nabla h_i(x^*) \\ \mu_j g_j(x^*) = 0 \quad j = 1, ..., m \\ i = (1, ..., k) \end{cases}$$

Les coordonnées du point x^* optimal au sens de pareto et les coefficients de Lagrange λ et μ satisfis donc les (n+m+p) équations :

$$\begin{cases} \sum_{j=1}^{m} \omega_j \frac{\partial f_j(x^*)}{\partial x_i} + \sum_{i=1}^{p} \mu_l \frac{\partial g_l(x^*)}{\partial x_i} + \sum_{i=1}^{p} \lambda_q \frac{\partial h_q(x^*)}{\partial x_i} \\ \mu_j g_j(x^*) = 0 \quad j = 1, ..., m. \\ h_l(x^*) = 0 \quad i = (1, ..., p). \\ i = (1, ..., k) \end{cases}$$

Il existe des poids $\omega_1^{'}, ..., \omega_k^{'}$ (qui dépendent de x^*) tels que l'on peut appliquer le théorème de Fritz John mono -objectif à le Fonction pondérée $f_{\omega'}(x) = \sum \omega_i^{'} f_i(x)$ le théorème apparaitre un coefficient λ_0 tel que $\lambda_0 \omega_i^{'} = \omega_i$ pour $i = 1, ..., k$.On revient alors à de l'optimisation mono-objectif cependant le théorème donne simplement l'existence de ces poids sans donner de moyen de les trouver ,la première égalité peut s'écrire à l'aide des différentielles des fonctions :

$df_{x^*}^T(\omega) + dg_{x^*}^T(\mu) + dh_{x^*}^T(\lambda) = 0.$

2.7.2 Conditions suffisantes d'optimalité

pour que les solution du système d'équation obtenues grâce au théorème de Fritz John soient les solution du problème d'optimisation multi-objectif sous contraintes , les fonctions coût et les contraintes doivent être convexes .on obtient alors le théorème de Karush-Kuhn et Tucker :

Théorème de Karush-Kuhn-Tucker [49]

Soient $f : \mathbb{R}^n \longrightarrow \mathbb{R}^k$,$g : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ et $h : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ des fonctions de classe C^1 .et telles que les fonctions f_i, g_j, h_l sont convexes .s'il existe $\omega \in \mathbb{R}^k_+, \lambda \in \mathbb{R}^p_+, \mu \in \mathbb{R}^m$ tels que

$$\begin{cases} \sum_{i=1}^{k} \omega_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) + \sum_{i=1}^{k} \lambda_l \nabla h_i(x^*) \\ \mu_j g_j(x^*) = 0 \\ \mu_j \geq 0 \quad j = 1, ..., m. \end{cases}$$

alors x est solution du problème multi-objectif :

$$(\mathrm{p}) : \begin{cases} optimiser f(x) \\ h(x) = 0 \\ g(x) \leq 0 \\ ou \\ x \in \mathbb{R}^n, f(x) \in \mathbb{R}^k, g(x) \in \mathbb{R}^m, h(x) \in \mathbb{R}^p \end{cases}$$

puisque dans ce cas , les fonctions cout $f_i$et les contraintes sont convexes , le minimum x^*est un minimum global du problème d'optimisation.

2.8 Méthodes de résolution des problèmes multi-objectif :

Les méthodes d'optimisation multicritères s'intéressent aux problèmes essayant d'optimiser simultanément plusieurs objectifs. Elles peuvent être classées selon le schéma suivant [15]

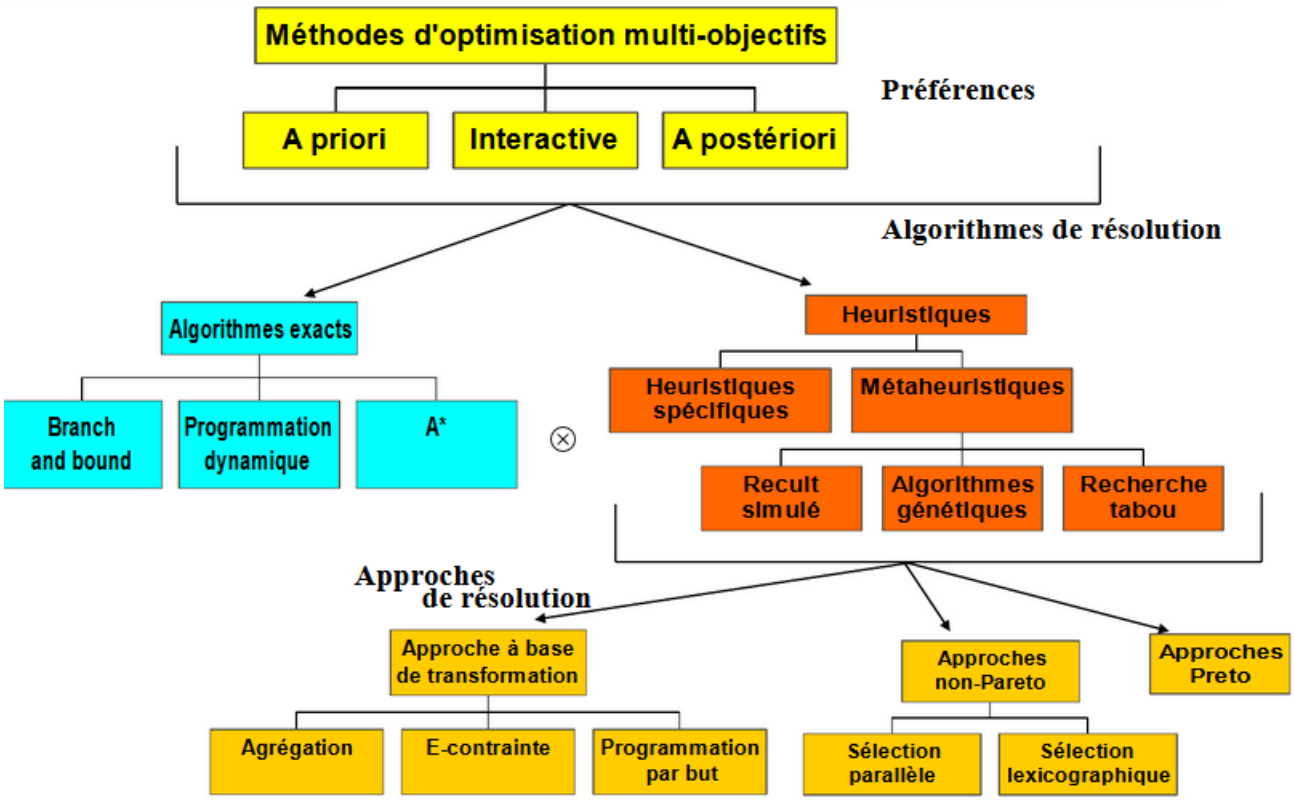


FIGURE 2.15 – :Classification des Méthodes d'optimisation multi-objectif.

Selon le mode de coopération entre le solveur et le décideur, elles peuvent être rangées en trois familles :

• Les méthodes la préférence a priori :

Dans ces méthodes, le décideur définit le compromis qu'il désire réaliser (il fait part de ces préférences) avant de lancer la méthode d'optimisation. Généralement, les fonctions objectifs et les critères de choix sont regroupes en une seule fonction objectif par agrégation. Il s'agit alors d'optimiser une fonction unique ou chaque objectif est considéré comme étant une variable d'optimisation, le problème est ainsi ramené a un problème d'optimisation mono-objectif. L'avantage de cette méthode est sa flexibilité dans le choix d'un algorithme d'optimisation mono-objectif.

De plus, si la méthode d'agrégation et les pondérations sont bien choisit, alors la méthode aura l'avantage d'être rapide la exécuter puisqu'elle ne nécessite qu'une seule exécution pour obtenir le compromis souhaite.

• Les méthodes la préférence progressives :

Dans ces méthodes, l'utilisateur affine son choix de compromis au fur et la mesure du déroulement de l'optimisation. Elles permettent donc de bien prendre en compte les préférences du décideur, mais nécessite sa présence tout au long du processus de recherche.On retrouve dans cette famille les méthodes interactives.

• Les méthodes la préférence a posteriori :

Cette méthodologie vise l'obtention d'un ensemble de solutions optimales parmi lesquelles le décideur effectue un choix. Aucun critère de préférence n'est alors incorpore dans l'étape d'optimisation. Cette étape vise ainsi l'obtention des solutions qui constituent les meilleurs compromis entre les différents objectifs.

Le principal avantage de cette méthode est son potentiel d'analyse des compromis obtenus permettant d'accroître la connaissance et la compréhension du problème la optimiser.

Les algorithmes de résolutions des PMOs se répartissent en deux grandes catégories :

2.8.1 Méthodes exactes

Les méthodes exactes cherchent à trouver de manière certaine la solution optimale en examinant de manière explicite ou implicite la totalité de l'espace de recherche. Elles ont l'avantage de garantir la solution optimale néanmoins le temps de calcul nécessaire pour atteindre cette solution peut devenir très excessif en fonction de la taille du problème (explosion combinatoire) et le nombre d'objectifs à optimiser. Ce qui limite l'utilisation de ce type de méthode aux problèmes bi-objectifs de petites tailles. ces méthodes génériques sont : Branch Bound, Branch Cut, Branch Price, Branch, Cut Price. d'autres méthodes sont moins générales, comme : La programmation dynamique, simplexe, La programmation linéaire en nombre entiers, L'algorithme A*. D'autres méthodes sont spécifiques à un problème donné comme l'algorithme de Johnson pour l'ordonnancement.

A) Branch Bound

Est une méthode générique de résolution exacte de problèmes d'optimisation, et plus particulièrement d'optimisation combinatoire. C'est une méthode constructive de recherche arborescente qui utilise l'énumération implicite basée sur la notion de bornes afin d'éviter l'énumération de larges

classes de mauvaises solutions. Elle utilise la stratégie diviser pour régner, en se basant sur deux concepts : le branchement (séparation) qui consiste à partitionner ou diviser l'espace des solutions en sous problèmes pour les optimiser chacun individuellement ; et l'évaluation qui consiste à déterminer l'optimum de l'ensemble des solutions réalisables associé au nœud en question ou, au contraire, de prouver mathématiquement que cet ensemble ne contient pas de solution optimale, la méthode la plus générale consiste à borner le coût des solutions contenues dans l'ensemble. Les techniques les plus classiques pour le calcul de bornes sont basées sur l'idée de relaxation de certaines contraintes : relaxation continue, relaxation lagrangienne...

B) La programmation dynamique

En informatique, la programmation dynamique est une méthode algorithmique pour résoudre des problèmes d'optimisation. Le concept a été introduit au début des années 1950 par Richard Bellman 1. à l'époque, le terme « programmation » signifie planification et ordonnancement1. La programmation dynamique consiste à résoudre un problème en le décomposant en sous-problèmes, puis à résoudre les sous-problèmes, des plus petits aux plus grands en stockant les résultats intermédiaires. Elle a d'emblée connu un grand succès, car de nombreuses fonctions économiques de l'industrie étaient de ce type, comme la conduite et l'optimisation de procédés chimiques, ou la gestion de stocks1.

C) Les algorithmes A*

En informatique, plus précisément en intelligence artificielle, l'algorithme de recherche A^{*2} (qui se prononce A étoile, ou A star à l'anglaise) est un algorithme de recherche de chemin dans un graphe entre un nœud initial et un nœud final tous deux donnés1. En raison de sa simplicité il est souvent présenté comme exemple typique d'algorithme de planification, domaine de l'intelligence artificielle. L'algorithme A^* a été créé pour 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 (en), Nils Jhon Nilsson et Bertram Raphael en 19682. il s'agit d'une extension

de l'algorithme de Dijkstra de 1959 .

2.8.2 Les heuristiques

Heuristiques spécifiques :

Elles sont dédiées à la résolution d'un PMO spécifique et ne fonctionnent que sur ce type de problèmes.

Méta-heuristique :

Elles sont fondées sur une idée générale. Et peuvent s'adapter à n'importe quel type de problème et donner de bons résultats. On trouve dans cette classe les algorithmes génétiques, la recherche tabou, le recuit simulé,... Les métaheuristiques résolvant les PMO peuvent être classées en trois approches :

Transformation des PMO en uni-objectif

≪ ces approches sont de type a priori≫

• Méthodes d'agrégation :[47]

Cette méthode est la plus largement utilisée en pratique et la plus efficace du point de vu informatique car elle est très facile à implémenter. D'ailleurs, elle est l'une des premières méthodes utilisées pour résoudre les problèmes multi-objectifs [53], elle est appelée aussi méthode de la somme pondérée.

Elle consiste à ramener le problème multi-objectifs en un problème mono-objectif. Elle somme tous les objectifs en affectant à chacun d'entre eux un coefficient de poids. Le décideur choisit les poids en fonction de l'importance relative des objectifs. Le problème multi-objectif (MOP) devient sous la forme suivante :

$$\text{Minimiser} \begin{cases} \sum \lambda_i f_i \quad combinaisonlinaire \quad des \quad f_i \\ avec \quad x \in \Omega \end{cases}$$

où :

$\lambda_i \geq 0$,$\forall i = 1,..,k$ et $\sum_{i=1}^{k} \lambda_i = 1$

λ_i : Appelle le poids, est une pondération associée au critère. Cette pondération permet d'exprimer des préférences sur les critères de décision.

La résolution du problème multi-objectifs (MOP) consiste à résoudre le problème paramétrique (2.16). La solution obtenue dépend fortement des coefficients choisis pour les poids. Cette méthode a été développée pour la généralisation des solutions non dominées. Elle est très efficace et applicable pour produire une solution non dominée initiale est peut être employé comme solution initiale pour d'autre technique.

Théorème

Toute solution optimale du problème est une solution Pareto optimale pour le problème multi-objectifs (MOP) si tous le poids sont strictement positifs.

Théorème [31]

Si le problème multi-objectifs (MOP) est convexe et si x^* est une solution Pareto optimale alors $\exists\ \lambda = (\lambda_1, \lambda_2, ..\lambda_k)$,tel que x^* est une solution optimale pour (p_λ)

• Programmation par but :

Cette méthode est également appelée "Target Vector Optimization". Charnes et Cooper proposent en 1961, cette méthode dans laquelle le décideur fixe un but T_i, $i = 1,..,k$ à atteindre pour chaque objectif f_i [53]. 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.

$$\text{Minimiser} \begin{cases} \sum |f_i(x) - T_i| \\ x \in \Omega \end{cases}$$

T_i : représente la valeur à atteindre pour le i^{me} objectif.

Remarque

-Cette méthode a l'avantage de fournir un résultat même si un mauvais choix initial a conduit le décideur 'a donner un ou plusieurs but(s) non réalisable(s).

- La méthode est facile ‘a mettre en œuvre mais la définition des buts ‘a atteindre est une question délicate.

• méthodesε-contraintes :

Cette méthode est probablement l'une des méthodes les plus connues pour la résolution des problèmes multi-objectifs. Elle a été introduite par Haimes en 1971 [55] et depuis elle est vue comme un outil tr‘es utilisé dans la résolution de ce type de problème, où elle transforme le problème multi-objectifs (MOP) en un problème mono-objectif. Elle est basée sur la minimisation d'un objectif jugé par le décideur prioritaire et de convertir les autres objectifs en contraintes supplémentaires au problème.

Le problème (P_ε) s'écrit comme suit :

$$\begin{cases} Minimiser f_i(x) \\ Telque : f_i(x) \leq \varepsilon_i \\ \forall i \in [0 \quad 1] \\ i \neq k \\ x \in \Omega \end{cases}$$

Théorème[56]

Pour tout vecteur $\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_k)$ tel que le problème p_ε a une solution optimale alors cette solution est Pareto optimale pour le problème (MOP).

Remarque

— Les poids λ_i et les paramètres ε_i sont choisis par le décideur.

— En faisant varier les poids et les paramètres dans les deux méthodes, on peut générer un sous ensemble de l'ensemble Pareto du problème multi-objectifs.

Exemple illustratif

Soit le problème multi-objectif suivantes :

$$\begin{cases} minf(x) = (f_1(x) = -3x_1 + x_2, f_2(x) = x_1 + x_2, f_3(x) = 2x_1 + x_2) \\ 3x_1 - x_2 \leq 6 \\ x_2 \leq 2 \\ x = (x_1, x_2) \in \mathbb{R}^2_+ \end{cases}$$

Résoudre le problème en considérant le critère 3 comme prioritaire avec
$\varepsilon = (\varepsilon_1, \varepsilon_2)^{'} = (1,3)^{'}$

$$\begin{cases} minf(x) = 2x_1 + x_2 \\ 3x_1 - x_2 \leq 6 \\ x_2 \leq 2 \\ (-)3x_1 + x_2 \leq 1 \\ x_1 + x_2 \leq 3 \\ x \in \mathbb{R}^2_+ \end{cases}$$

La résolution de p_ε par la méthode du simplexe nous donne la solution optimale
$x^* = (x_1, x_2) = (0,0)^t$
Donc, cette solution est Pareto optimale pour le problème initial d'après le théorème .

2.8.3 Approches non Pareto :

≪ ces approches sont de type a posteriori ≫
Ces approches ne transforment pas le problème multiobjectif 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.

Sélection parallèle

Cette approche a été la première proposant un algorithme génétique pour la résolution de problèmes multiobjectifs [21]. 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).

Sélection lexicographique :

Cette approche proposée par Fourman [39] 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.

• **Discussion :**

L'inconvénient majeur de cette méthode est qu'elle requiert un choix de la séquence des objectifs à minimiser. Deux choix différents génèrent deux solutions distinctes.

2.8.4 Approches Pareto :

≪ ces approches sont de type a posteriori ≫

Ces approches utilisent la notion de dominance lors de la sélection des solutions. Elles sont capables de générer des solutions Pareto situées dans des concavités. Historiquement, on peut considérer l'existence de deux générations des MOEAs. la première génération des approches Pareto est caractérisée par l'utilisation :

• d'un mécanisme de sélection Pareto. En effet la relation de dominance permet d'affecter des rangs aux individus de la population, ce qui fait apparaître la notion de front. C'est la technique de ranking [8].

• du sharing (niching) qui consiste à pénaliser la valeur d'adaptation en fonction du nombre d'individus au voisinage du regroupement cette technique introduite dans [10] est largement utilisée maintenant. Les algorithmes correspondants sont relativement simples. Parmi les approches les plus représentatives de cette génération on peut citer :

• Multi-Objective Genetic Algorithm (M.O.G.A)

• N.S.G.A (Non dominated Sorting Genetic Algorithm)

• N.P.G.A (Niched-Pareto Genetic Algorithm)

Après environs dix années de succès, cette génération devient dépassée. La deuxième génération des approches Pareto est maintenant considérée comme l'état d'art de l'optimisation multi-objectif. Ce sont en général des approches élitistes manipulant une population secondaire externe. Parmi les approches les plus représentatives de cette génération on peut citer :

• SPEA et SPEA2 (Strength Pareto Evolutionary Algorithm)

• NSGA-II (Non dominated Sorting Genetic Algorithm)

• PAES (The Pareto Archived Evolution Strategy)

• PESA et PESA II (The Pareto Envelope-based Selection Algorithm)

• MOMGA et MOMGA II (Multi-objective Messy Genetic Algorithm)

• MicroGA pour l'Optimisation multi-objectif.

Comme les métaheuristiques ont permis la réalisation de méthodes de résolutions très performantes ,on a préféré les aborder avec un peu plus de détails dans la section qui suit.

2.9 Différents méthodes de résolutions des problèmes d'optimisation multi-objectifs

De nombreuses méthodes d'optimisation multi-objectifs existent. Un grand nombre de ces méthodes ont et le recenseuses et décrites dans l'ouvrage de Colette et Siarry .

Les premières tendances aboutissent la des méthodes purement mathématiques dont elles ne peuvent pas tendre vers l'ensemble de solutions dans un temps optimal et elles sont compliquées, par exemple, la programmation linéaire, la méthode de Simplexe, ...etc . on outre, une deuxième tendance a pour but d'avoir des méthodes qui tendent vers l'ensemble de solutions dans un temps optimal et d'une manière simple en se basant sur le principe des heuristiques, cette tendance est " les meta-heuristiques ", comme Le recuit simule, Colonies de fourmis ... etc.

Pour classer ces méthodes, plusieurs classement sont apparut selon des critères différents, quelques auteurs les classifient en cinq groupes :

-Les méthodes scalaires.

- Les méthodes interactives.

-Les méthodes floues.

-Les méthodes d'aide la la décision.

-Les méthodes exploitant une méta-heuristiques.

Ces méthodes de ces quatre groupes peuvent aussi être rangées en trois familles de d'optimisation multi-objectifs [7]

2.9.1 Méthodes scalaires

Ces méthodes sont dans la familles la préférence a posteriori. A l'origine, les problèmes multi-objectifs étaient transformes en problèmes mono-objectif, plusieurs approches différentes ont et le mises au point pour se faire, telles que : **La méthode de pondération de fonctions objectives. La méthode de Keeney-Raiffa. La méthode de la distance à un objectif de référence. La méthode de compromis.**

2.9.2 Méthodes interactives

Les méthodes interactives permettent de chercher une et une seule solution. Elles forment la famille des méthodes progressives et permettent à l'utilisateur de déterminer ses préférences vis-à-vis d'un compromis entre objectifs au cours de l'optimisation. Nous allons maintenant présenter quelques méthodes interactives (ou $plutôt$ quelques principes d'interaction).

La méthode de compromis par substitution :

Cette méthode (Surrogate Worth Tradeoff (SWT) ou « méthode de compromis par substitution » a été très utilisée pour l'optimisation de ressource en eau [57]. Elle est basée sur la méthode de compromis, à laquelle on a ajouté un processus interactif, pour que la méthode converge vers la solution la plus susceptible de satisfaire l'utilisateur. [53]

Méthode de Jahn :

Cette méthode est complètement différente. on commence par choisir un point de départ, puis le problème est résolu pas à pas à travers l'espace de recherche en direction de la solution optimale au sens de Pareto [20].

Cette méthode est fondée sur une méthode de minimisation cyclique (aussi appelée « méthode

d'optimisation scalaire dans les directions réalisables » de zoutendjik).

Méthode de geoffrion :

Cette approche similaire à la méthode de John. elle repose sur un algorithme nommé : l'algorithme de Frank-Wolfe [20].

2.9.3 Méthodes floues

Dans la vie courante, tout ne peut être décrit de manière binaire. Par exemple, la transition entre le jour et la nuit se fait progressivement, l'action sur l'embrayage d'un véhicule est, elle aussi, progressive.

Longtemps, le seul outil de description en logique était binaire. Tout en logique a été décrit en termes de VRAI ou FAUX. Le problème de cette description simpliste en logique et qu'elle ne permet pas de traiter l'incertitude de l'imprécision des connaissances humaines.

$L'automaticien\ L.A.\ Zadeh$ a élaboré une nouvelle logique basée sur les ensembles flous. Elle permet de traiter l'imprécision et l'incertitude dans la connaissance humaine ainsi que les transitions progressives entre états.

La différence principale entre une logique classique et cette logique floue est l'existence d'une transition progressive entre le VRAI et le FAUX [32].

Méthode de Sakawa

Cette méthode fait intervenir la logique floue à tous les niveaux (sur les paramètres du problème ainsi que sur les paramètres des contraintes). L'ensemble de solutions que l'on va trouver, lui aussi, fera intervenir la logique floue. Ces solutions auront un niveau d'appartenance. Ce seront des solutions qui auront une corrélation variable avec l'objectif initial [40] (le niveau de corrélation avec l'objectif initial est fixé par l'utilisateur)[53].

2.9.4 Méthodes exploitant une méta-heuristique

Les méthodes classiques de résolution des problèmes d'optimisation ne sont pas adaptées à tout type de problème dont les fonctions à optimiser sont des fonctions difficiles ou complexes.
Les nouvelles méthodes sont souvent inspirées de mécanismes d'optimisation rencontrées dans la nature. Elles sont incomplètes dans le sens ou elles n'explorent délibérément qu'une partie de l'espace de recherche, elles sont utilisées pour les problèmes où on ne $connâit$ pas d'algorithmes de résolution en temps polynomial et pour lesquels on espère trouver une solution approchée de l'optimum global. Ces méthodes utilisent généralement des heuristiques ou des méta-heuristiques pour ce guider dans l'espace de recherche et choisir les solutions proche de l'optimum, Plusieurs classifications des méta-heuristiques ont été proposées, la plupart distinguent globalement deux catégories : celles se basant sur une solution unique, et celles se basant sur une population de solution.

2.9.5 Méthodes d'aide à la décision

Les méthodes que nous avons présentées jusqu'ici étaient basées sur la relation de dominance. Cette relation (que l'on peut définir de plusieurs manières : la dominance de Pareto, la dominance lexicographique par exemple) permet de filtrer les éléments d'un ensemble, et de ne retenir que les éléments incomparables entre eux. Cependant, il existe une autre approche pour obtenir un ensemble de solutions, qui repose sur l'établissement d'une relation d'ordre entre les différents éléments. Ainsi, on peut, en fonction de la relation d'ordre définir , obtenir un ensemble de solutions (relation d'ordre partiel) ou une et une seule solution (ordre complet). L'autre différence majeure, par rapport aux méthodes d'optimisation multi-objectifs "classiques", vient du fait que les méthodes d'aide à la décision ne travaillent que sur des ensembles discrets de points (les méthodes d'optimisation multi-objectifs "classiques" peuvent travailler, elles, sur des ensembles continus).
Parmi les méthodes d'aide à la décision les plus utilisées, on ite : les méthodes ELECTRE (I, IS, II, III, IV et TRI), et les méthodes PROMETHEE (I et II). Chacune de ces méthodes traite une problématique bien précise. Les éléments importants qui différencient ces méthodes sont la définition

de la relation de classement des actions (chaque méthode exploite une relation de préférence différente) et la méthode d'exploitation des résultats.
Une autre méthode d'aide à la décision qui aide le décideur dans la pondération des critères, est la méthode AHP, nous donnerons les détailles et les étapes de calcul dans la section suivante.

Chapitre 3

MÉTA-HEURISTIQUE ET ALGORITHMES GÉNÉTIQUES

3.1 Définition de la simulation :

La simulation est une technique numérique pour conduire des expériences sur un ordinateur qui peut inclure des caractéristiques stochastiques soit microscopiques soit macroscopiques et implique l'utilisation de modèles mathématiques qui décrivent le comportement d'un système de transport sur des périodes étendues de temps réel.
La simulation de circulation peut servir dans plusieurs cas :

- Choisir entre différentes alternatives de conception
- Etudier la sécurité routière
- Tester une nouvelle conception Raffiner une nouvelle conception
- Etre intégré dans d'autres outils (pour l'étude d'optimisation d'un feu par exemple)
- Entraîner le personnel de surveillance de réseau.

Différents méthodes de la simulation

La méthode de Monte Carlo [38]

Les méthodes de Monte-Carlo qu'ont pour essence l'utilisation d'expérience répétées pour evaluer une quantité résoudre un système déterministe.
Ces méthodes peuvent servir pour :

— Le calcul d'intégrale

— La résolution d'équation aux dérivées partiels

— La résolution de système linéaire

— La résolution de problème d'optimisation

Considérons par exemple le problème de l'intégration numérique .
Il s'agit d'approche $I = \int_0^1 g(x)dx$ divères méthodes classiques de type déterministe existent :Trapèze,Simpson,rectangle et I est approximé par une somme de la forme $\sum_{i=0}^{n} w_i g(x_i)$ avec $\sum_{i=0}^{n} w_i = 1$ et $(x_i)_{0<i<1}$ bien choisis dans [0 1].
La méthode de monte carlo consiste à écrire sous la forme $I = E(g(u))$ où u est une variable aléatoire suivent la loi $u \rightsquigarrow U[01]$ et à utiliser la loi des grandes nombres si $(U_i)_{0<i<n}$ est une suite de variable aléatoire indépendant et de la loi uniforme sur [0 1] ,alors $\frac{1}{n}\sum_{i=1}^{n} g(u_i) \longrightarrow E(g(u))$.
En d'autres termes si $u_1, u_2, ..., u_n$ sont des nombres tires au hasard dans [0 1] alors $\frac{1}{n}(g(u_1) + g(u_2) + ... + g(u_n))$ est une approximation de $\int_0^1 g(x)dx$ prenons l'exemple le plus générale d'une intégrale de type $I = \int_{\mathbb{R}^d} g(x)f_x(x)dx$ où $f_x(x) \geq 0$ et $\int_{\mathbb{R}^d} f_x(x)dx = 1$
Alors $I = E(g(x))$ où x est une variable aléatoire de densité f par rapport à la mesure de lebesgue sur $\mathbb{R}^d$.
Toujours par la loi des grandes nombres si $(x_i)_{i\in\mathbb{N}}$ est une suite des variables aléatoires indépendants sur $\mathbb{R}^d$ de la loi de densité f $\frac{1}{n}\sum_{i=1}^{n} g(u_i) \longrightarrow E(g(u))$ et donc si $(x_1, x_2, ..., x_n)$ est une réalisation de $(x_1, x_2, ..., x_n)$,$\frac{1}{n}(g(x_1) + g(x_2) + ... + g(x_n))$ sera une approximation de I.

Méthode d'inversion de la fonction de répartition

Les transformations de variable nous permettons de simuler des échantillons fictifs d'une variable aléatoire à partir d'un ensemble des nombres aléatoires uniforme distribués sur [0 1].la méthode de la transformation inverse peut utiliser pour générer n'importe quelle variable aléatoire à condition que on connaisse analytiquement la fonction de répartition inverse F^{-1} .

Théorème :

Soit x une variable aléatoire de fonction de répartition F comme si u est une variable uniforme sur [0 1] alors $F^{-1}(u)$ à même loi que x.

Remarque :

Une application directe de ce théorème peut à partir d'une loi uniforme de générer n'importe quelle autre loi.ainsi pour générer une échantillon fictif issu d'une variable aléatoire x il faut connaitre F et avoir une suite de nombre $y_1, y_2, ..., y_n$ issus d'une variable aléatoire u uniforme sur [0 1] l'égalité $x = F^{-1}(y)$ peut d'obtenir l'échantillon $x_1 = F^{-1}(y_1), x_2 = F^{-1}(y_2), ..., x_n = F^{-1}(y_n)$ issu d'une population distribués selon F. La seule difficulté est celle de calcule F^{-1} .

3.2 Heuristique

En optimisation combinatoire, une heuristique est un algorithme approché qui permet d'identifier en temps polynomial au moins une solution réalisable rapide, pas obligatoirement optimale. L'usage d'une heuristique est efficace pour calculer une solution approchée d'un problème et ainsi accélérer le processus de résolution exacte. Généralement une heuristique est conçue pour un problème particulier, en s'appuyant sur sa structure propre sans offrir aucune garantit quant à la qualité de la solution calculée. Les heuristiques peuvent être classées en deux catégories :

— Méthodes constructives qui génèrent des solutions à partir d'une solution initiale en essayant d'en ajouter petit à petit des éléments jusqu'à ce qu'une solution complète soit obtenue,

— Méthodes de fouilles locales qui démarrent avec une solution initialement complète (probablement moins intéressante), et de manière répétitive essaie d'améliorer cette solution en explorant son voisinage.

3.3 Méthodes de trajectoire

Les méthodes de recherche locale passent d'une solution à une autre dans l'espace des solutions candidates (l'espace de recherche) qu'on note S, jusqu'à ce qu'une solution considérée comme optimale soit trouvée ou que le temps imparti soit dépassé. La méthode de recherche locale la plus élémentaire est la méthode de descente.

La méthode de descente

Pour un problème de minimisation d'une fonction f , la méthode descente peut être décrite comme suit :

Algorithme La méthode de descente

1) **Solution initiale s ;**
2) **Répéter :**
3) **Choisir** $s^{'}$ **dans un voisinage N(s) de s ;**
4) **Si** $f(s^{'}) < f(s)$ **alors** $s := s^{'}$ **;**
5) **jusqu'à ce que** $f(s^{'}) > f(s)$ **,** $\forall s^{'} \in N(s)$**.**

La figure présente l'évolution d'une solution dans la méthode de descente.

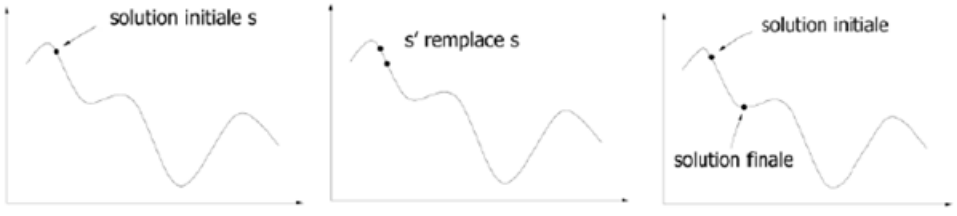


FIGURE 3.1 – .Evolution d'une solution dans la méthode de descente

3.4 Méta-heuristique

Face aux difficultés rencontrées par les heuristiques pour avoir une solution réalisable de bonne qualité pour des problèmes d'optimisation difficiles, les métaheuristiques ont fait leur apparition. Ces algorithmes sont plus complets et complexes qu'une simple heuristique, et permettent généralement d'obtenir une solution de très bonne qualité pour des problèmes issus des domaines de la recherche opérationnelle ou de l'ingénierie dont on ne connait pas de méthodes efficaces pour les traiter ou bien quand la résolution du problème nécessite un temps élevé ou une grande mémoire de stockage.

Le rapport entre le temps d'exécution et la qualité de la solution trouvée d'une métaheuristique reste alors dans la majorité des cas très intéressant par rapport aux différents types d'approches de résolution.

La plupart des métaheuristiques utilisent des processus aléatoires et itératifs comme moyens de rassembler de l'information, d'explorer l'espace de recherche et de faire face à des problèmes comme l'explosion combinatoire. Une métaheuristique peut être adaptée pour différents types de problèmes, tandis qu'une heuristique est utilisée à un problème donné.Plusieurs d'entre elles sont souvent inspirées par des systèmes naturels dans de nombreux domaines tels que : la biologie (algorithmes évolutionnaires et génétiques) la physique (recuit simulé), et aussi l'éthologie(algorithmes de colonies de fourmis).

Un des enjeux de la conception des métaheuristiques est donc de faciliter le choix d'une méthode et le réglage des paramètres pour les adapter à un problème donné.

Les métaheuristiques peuvent être classées de nombreuses façons. On peut distinguer celles qui travaillent avec une population de solutions de celles qui ne manipulent qu'une seule solution à la fois. Les méthodes qui tentent itérativement d'améliorer une solution sont appelées méthodes de recherche locale ou méthodes de trajectoire. Ces méthodes construisent une trajectoire dans l'espace des solutions en tentant de se diriger vers des solutions optimales. Les exemples les plus connus de ces méthodes sont : La recherche Tabou et le Recuit Simulé. Les algorithmes génétiques,l'optimisation par essaim de particules et Les algorithmes de colonies de fourmis présentent les exemples les plus connus des méthodes qui travaillent avec une population.

3.4.1 Le Recuit Simulé (simulated annealing)

Le recuit simulé (SA) a été introduit par [50, 51] comme une méthode de recherche locale normale, utilisant une stratégie pour éviter les minima locaux. Cette métaheuristique est basée sur une technique utilisée depuis longtemps par les métallurgistes qui, pour obtenir un alliage sans défaut, faisant alterner les cycles de réchauffage (ou de recuit) et de refroidissement lent des métaux. Le recuit simulé s'appuie sur des travaux faites par [43], qui ont pu décrire l'évolution d'un système en thermodynamique.

[13] Le principe du recuit simulé est de parcourir de manière itérative l'espace des solutions. on part avec une solution notée s_0 initialement générée de manière aléatoire dont correspond une énergie initiale E_0 et une température initiale T_0 généralement élevée. a chaque itération de l'algorithme, un changement élémentaire est effectué sur la solution, cette modification fait varier l'énergie du système $\triangle E$. si cette variation est négative (la nouvelle solution améliore la fonction objective, et permet de diminuer l'énergie du système), elle est acceptée. si la solution trouvée est moins bonne que la précédente alors elle sera acceptée avec une probabilité P calculée suivant la distribution de Boltzmann suivante :

$$P(E,T) = \exp^{\frac{\triangle E}{T}}$$

En fonction du critère de Metropolis, un nombre $\epsilon \in [0\ 1]$ est comparé à la probabilité $p = \exp^{\frac{\triangle E}{T}}$. Si $p < \epsilon$ la nouvelle solution est acceptée.

Le fonctionnement du critère de Metropolis est interprété par

- Si $\triangle E = f(s^{'}) - f(s) < 0$ alors $\exp^{\frac{\triangle E}{T}} > 1$, donc e est toujours inférieur à cette valeur, et on accepte la solution $s^{'}$
- Si $\triangle E > 0$
- et T est très grande, alors $\exp^{\frac{\triangle E}{T}} \cong 1$ tout voisin est systématiquement accepté.
- et T est très petite, alors $\exp^{\frac{\triangle E}{T}} \cong 0$ une dégradation a peu de chances d'être acceptée.

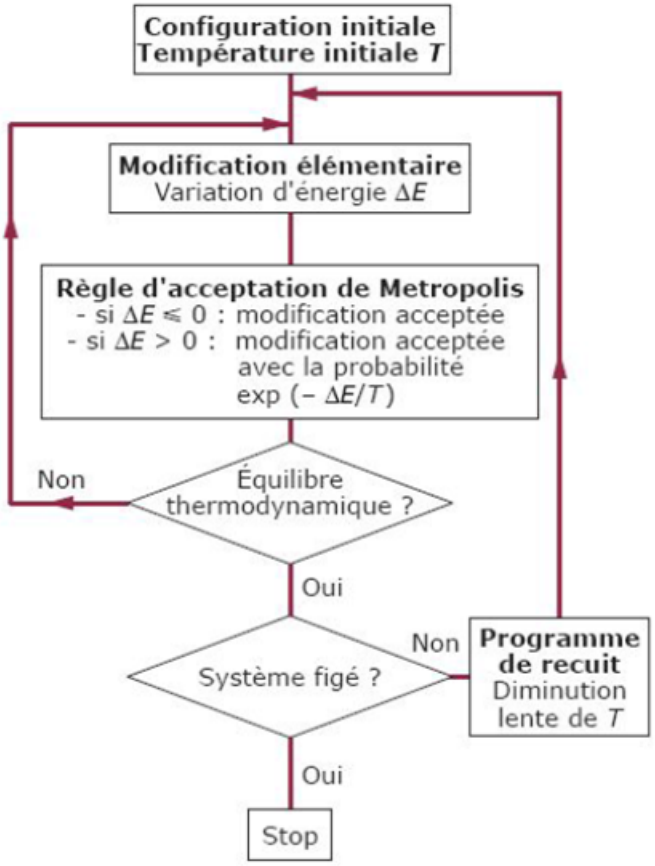


FIGURE 3.2 – .Fonctionnement de l'algorithme de recuit simulé.

Algorithme de Reçuit Simulé

1) Engendrer une configuration initiale S_0 de S : $S_0 \longrightarrow S$
2) Initialiser la température T en fonction du schéma de refroidissement
3) Répéter
4) Engendrer un voisin aléatoire S_0 de S
5) Calculer $\triangle E = f(s^{'}) - f(s)$
6) Si $\triangle E \leq 0$ alors $S^{'} \longrightarrow S$
7) Sinon accepter S_0 comme la nouvelle solution avec la probabilité $P(E,T) = \exp^{\frac{\triangle E}{T}}$
8) Fin si
9) Mettre T à jour en fonction du schéma de refroidissement (réduire la température)

10) Jusqu'à la condition d'arrêt

11) Retourner la meilleure configuration trouvée

Le choix de la température est primordial pour garantir l'équilibre entre l'intensification et la diversification des solutions dans l'espace de recherche. Premièrement, le choix de la température initiale dépend de la qualité de la solution de départ. Si cette solution est choisie aléatoirement, il faut prendre une température relativement élevée.
On utilise souvent la règle suivante :

$$T_k.\alpha \longrightarrow T_{k+1}$$

où $\alpha \in [0\ 1]$, un paramètre qui exprime la diminution de la température de l'itération k à k + 1. La décroissance de la température peut également être réalisée par paliers(en d'autres termes, elle ne change qu'après un certain nombre d'itérations).Certains préconisent l'utilisation de stratégies non monotones. On peut ainsi rehausser la température lorsque la recherche semble bloquée dans une région de l'espace de recherche. On peut alors considérer une grande augmentation de la température comme un processus de diversification alors que la décroissance de la température correspond à un processus d'intensification.

Avantages

— La méthode du recuit simulé a l'avantage d'être souple vis-à-vis des évolutions du problème et facile à implémenter.

— Contrairement aux méthodes de descente, SA évite le piège des optima locaux.

— Excellents résultats pour un nombre de problèmes complexes.

Inconvénients

— Nombreux tests nécessaires pour trouver les bons paramètres.

— Définir les voisinages permettant un calcul efficace de $\triangle E$.

3.4.2 La recherche Tabou (Tabu Search)

La recherche tabou (TS) est une méthode de recherche locale combinée avec un ensemble de techniques permettant d'éviter d'être piégé dans un minimum local ou la répétition d'un cycle.

La recherche tabou est introduite principalement par Glover [17], Hansen [46],Glover et Laguna dans [18]. Cette méthode a montré une grande efficacité pour la résolution des problèmes d'optimisation difficiles. En effet, à partir d'une solution initiale s dans un ensemble de solutions local S, des sous-ensembles de solution N(s) appartenant au voisinage S sont générés. Par l'intermédiaire de la fonction d'évaluation nous retenons la solution qui améliore la valeur de f , choisie parmi l'ensemble de solutions voisines N(s). L'algorithme accepte parfois des solutions qui n'améliorent pas toujours la solution courante. Nous mettons en oeuvre une liste tabou (tabu list) T de longueur k contenant les k dernières solutions visitées, ce qui ne donne pas la possibilité a une solution déjà trouvée d'être acceptée et stockée dans la liste tabou. Alors le choix de la prochaine solution est effectué sur un ensemble des solutions voisines en dehors des éléments de cette liste tabou. Quand le nombre k est atteint, chaque nouvelle solution sélectionnée remplace la plus ancienne dans la liste. La construction de la liste tabou est basée sur le principe FIFO, c'est-à-dire le premier entré est le premier sorti. Comme critère d'arrêt on peut par exemple fixer un nombre maximum d'itérations sans amélioration de s^*, ou bien fixer un temps limite après lequel la recherche doit s'arrêter.

Algorithme La recherche Tabou

1) Initialisation : s_0 une solution initiale
   $s_0 \longrightarrow s$, $s^* \longrightarrow s$,$f(s_0) \longrightarrow c^*$
   $T = \varnothing$
2) Générer un sous-ensemble de solution au voisinage de s
   $s^{'} \in N(s)$ tel que $\forall x \in N(s), f(x) \geq f(s^{'})$ et $s^{'} \notin T$
   Si $f(s^{'}) < c^*$ alors $s^{'} \longrightarrow s^*$ et $f(s^{'}) \longrightarrow c^*$
   Mise-à-jour de T
3) Si la condition d'arrêt n'est pas satisfaite retour à l'étape 2

Avantages et inconvénients

La recherche tabou est une méthode de recherche locale, et la structure de son algorithme de base est proche de celle du recuit simulé, avec l'avantage d'avoir un paramétrage simplifié : le paramétrage consistera d'abord à trouver une valeur indicative t d'itérations pendant lesquelles

les mouvements sont interdits. Il faudra également choisir une stratégie de mémorisation. En revanche, la méthode tabou exige une gestion de la mémoire de plus en plus lourde en mettant des stratégies de mémorisation complexe. L'efficacité de la méthode tabou offre son utilisation dans plusieurs problèmes d'optimisation combinatoire classiques tels que le problème de voyageur de commerce, le problème d'ordonnancement, le problème de tournéees de véhicules, etc.

3.5 Algorithmes génétiques

Définition :

Les algorithmes génétiques [25] 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 différentiabilité de la fonction à optimiser, ce qui est d'ailleurs un des points qui différencie les algorithmes génétiques des autres méthodes classiques. Les caractéristiques qui font l'unicité de cette méthode sont :

1) Les AGs ne travaillent pas directement sur les paramètres, ces derniers sont codés avant l'opération.
2) Ils opèrent sur une population de solutions au lieu d'une seule solution.
3) Les AGs n'utilisent que les valeurs de la fonction à étudier, sa dérivée ou autres connaissances ne sont pas prises en considération.
4) Les AGs se basent sur des données aléatoires, donc ils sont probabilistes.

Les AGs utilisent les éléments suivant pour se définir :

— Individu/chromosome/séquence : une potentielle solution du problème abordé.
— Population : un ensemble de chromosomes de l'espace de l'exploration
— Environnement : l'espace de l'exploration et de la recherche.
— Fonction d'évaluation (fitness) : la fonction objectif qui sert à évaluer la précision de la solution du problème.

3.5.1 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) Opérations génétiques : Croisement et mutation de certains individus au sein de la nouvelle population.
5) Retour : Tant que la condition d'arrêt n'est pas satisfaite, l'algorithme recommence à partir de la phase d'évaluation.

Un AG génère une population en utilisant une représentation chromosomique des individus 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 :

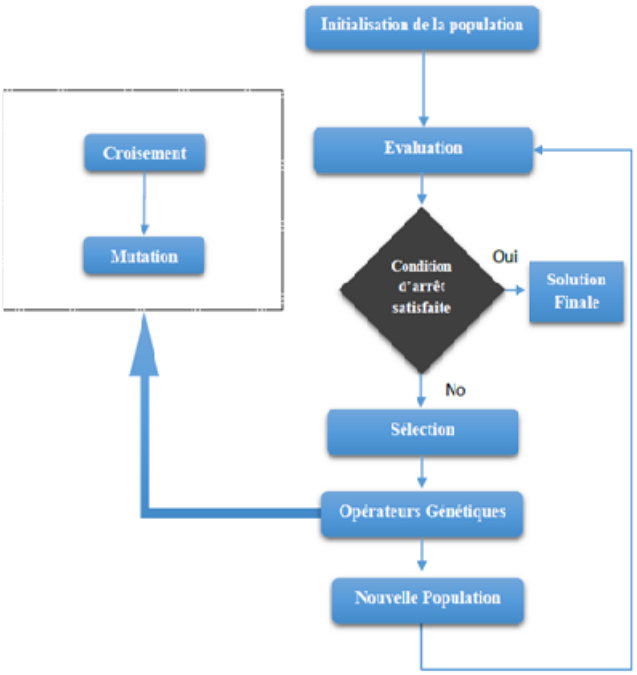


FIGURE 3.3 – Fonctionnement de l'algorithme génétique .

3.5.2 Représentation chromosomique(Codage)

Au début, chaque individu (chromosomique) de la population est codé soit en binaire/gray[8], ou en réel [9], représenté par un vecteur dont la longueur dépend de la précision requise, le codage donne à l'algorithme génétique une représentation génotype-phénotype.

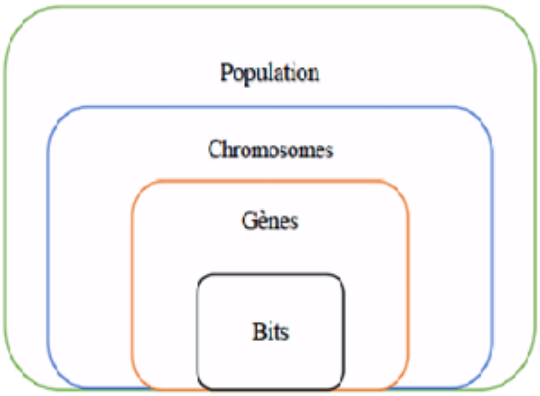


FIGURE 3.4 – Représentation chromosomique .

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.

Exemple de 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 :

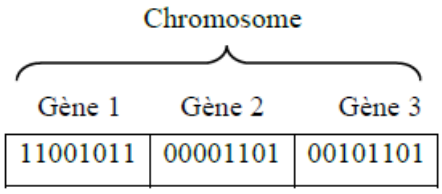


FIGURE 3.5 – Codage binaire de trois variables.

3.5.3 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.

3.5.4 Sélection

C'est dans cette étape où se décide quels sont les individus de la population qui vont 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) [8]

— La méthode de rang [23].

— La sélection par tournoi [11]

— La sélection universelle stochastique [24].

3.5.5 Opérateurs génétiques

Croisement

Pendant cette opération, la population est divisée en deux sous-populations, ensuite, 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 0.5 et 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 [8] et trois segments pour le croisement en deux points [30]. 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 nouveau chromosomes.

Les schémas ci-dessous illustrent les deux méthodes :

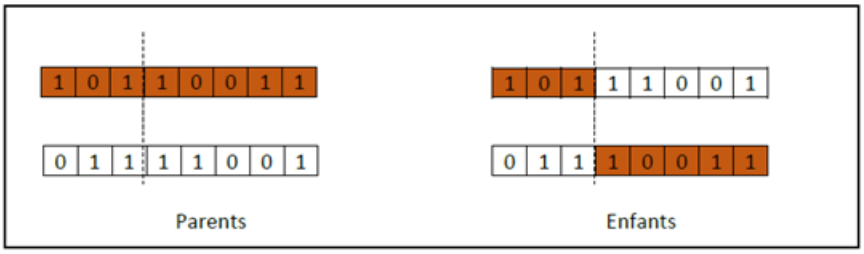


FIGURE 3.6 – Croisement en un seul point.

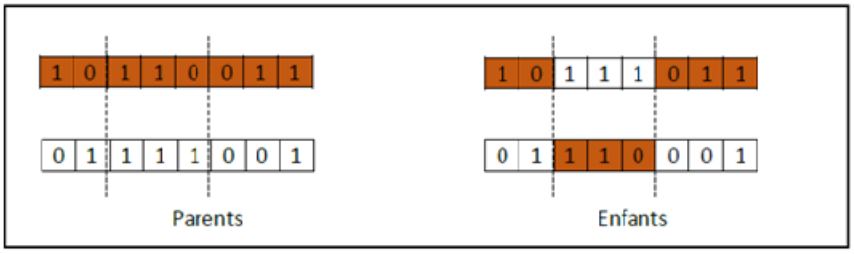


FIGURE 3.7 – Croisement en deux points.

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 individu a une probabilité $P_m$de subir une mutation, un réel $r \in \{01\}$ est généré, si $r < Pm$ le bit sera donc remplacé par complémentaire [8].

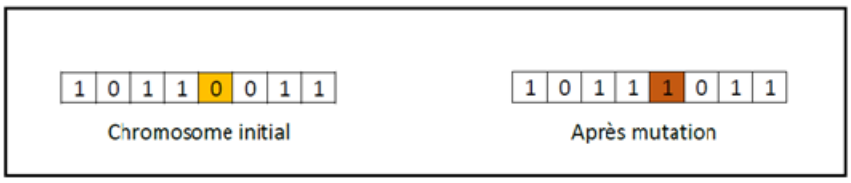


FIGURE 3.8 – Mutation d'un chromosome.

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

Chapitre 4

MÉTHODE NSGA-II (A Fast Multiobjective Genetic Algorithm)

4.1 NSGA-I

L'algorithme NSGA proposé par Deb et Srinivas [44] est semblable aux algorithmes génétiques qui ont été présentés à la section . Aucun choix n'est imposé sur les opérateurs génétiques de mutation, croisement, sélection et insertion. La seule différence se trouve au niveau de l'implantation de l'opérateur d'assignation de la valeur d'adaptation des individus.

Algorithme de la fonction de notation

- Dans la population entière, On recherche les individus non dominés. Ces derniers constituent la première frontière de Pareto.
- On leur attribue une fitness factice. Cette valeur est supposée donner une chance égale de reproduction à tous ces individus. Mais pour maintenir la diversité dans la population, il est nécessaire d'appliquer une fonction de sharing sur cette valeur.
- Ce premier groupe d'individus est supprimé de la population
- On recommence cette procédure pour déterminer la seconde frontière de Pareto. La valeur fac-

tice de fitness attribuée à ce second groupe est inférieure à la plus petite fitness après application de la fonction de sharing sur le premier groupe. Ce mécanisme est répété jusqu'à ce qu'on ait traité tous les individus de la population.

- L'algorithme se déroule ensuite comme un algorithme génétique classique. Srinivas utilise une sélection basée sur le reste stochastique. Mais sa méthode peut être utilisée avec d'autres heuristiques de sélections (tournoi,roulette,... etc).

Critique

Trois critiques ont été soulevées pour cette méthode [48] :

- Sa complexité de calcul de $O(K, N^3)$ avec k le nombre d'objectifs et N la taille de la population, essentiellement due au processus de trie de la population et d'application de l'heuristique de partage

- Son approche non élitiste.

- La nécessité de spécifier un paramètre de sharing.

Algorithme

Procédure d'assignation de la valeur d'adaptation du NSGA [44].

Données ρ :Population à trier.

$\mathscr{F}_k$: Sous-ensemble de $Pareto$.

A : Opérateur de fonction de partage.

T : Opérateur de tri par ensemble de $Pareto$.

ω_i : Valeur d'adaptation de la solution i.

σ_{share} :Paramètre servant à la gestion de la diversité.

ε : Petit nombre positif.

Ω_{min} : Valeur d'adaptation minimum des solutions d'un ensemble de $Pareto$.

z_k^i : Solution i faisant partie de l'ensemble de $Pareto$ k.

début Installer σ_{share} ,ε , $\Omega_{min} = \|\rho\| + \varepsilon$ et $k = 1$

Appliquer $\rho' = T(\rho)$

pour chaque $\mathscr{F}_k \in \rho'$ faire

pour chaque $z_k^i \in \mathscr{F}_k$ faire

Assigner $\omega_i = \Omega_{min} - \varepsilon$

Modifier $\omega_i = \mathbf{A}(z_k^i)$

Mettre à jour $\Omega_{min} = min(\omega_i, \forall i \in \mathscr{F}_k$

K=K+1

fin

Cette assignation repose sur le rang de l'ensemble de $Pareto$ dans lequel la solution est incluse. Dans la procédure d'assignation, toutes les solutions du premier ensemble auront comme valeur d'adaptation $\omega_i = \|\rho\| \quad \forall i \in \mathscr{F}_k$. Pour tous les autres ensembles la valeur d'adaptation sera égale à la plus petite valeur de la solution du front qui le précède retranché de ", un petit nombre positif. De cette façon, il n'y aura jamais deux solutions faisant partie de deux ensembles différents ayant la même valeur d'adaptation.Le principal désavantage de cet algorithme est l'absence d'élitisme. Les prochaines sections présenteront différents algorithmes qui permettent l'ajout de cet opérateur.

4.1.1 NSGA-II

L'algorithme NSGA-II est la version modifiée de NSGA par Deb et d'autres chercheurs ; après plusieurs années de développement et d'utilisation, l'algorithme NSGA-II est actuellement considéré comme un très bon et mature algorithme d'optimisation multi-objectifs. Dans l'algorithme NSGA-II, la fonction de distance d'encombrement est utilisée pour calculer la valeur de fitness de chaque individu dans l'algorithme. Selon l'idée de tri rapide non dominante et la sélection environnementale des PCD, l'algorithme utilise les deux stratégies pour faire évoluer la population évolutive afin d'optimiser le problème correspondant. Dans le reste de cet article, trois aspects importants de cet algorithme sont introduits pour étudier le NSGA-II original : le tri rapide non dominant, qui est utilisé pour évaluer chaque individu dans l'algorithme NSGA-II ; fonction de distance d'encombrement qui est utilisée pour calculer la valeur de forme physique de chaque individu. La sélection environnementale basée sur la PCD est utilisée pour sélectionner les individus à la prochaine génération de la population évolutive dans l'algorithme

Fast Nondominated Sorting (FNS)

Dans l'algorithme d'optimisation évolutive multiobjectifs, l'évolution de la population consiste à éliminer la solution inférieure et à sélectionner les excellentes solutions par une certaine stratégie, de sorte que la solution inférieure soit éliminée et que l'excellente solution soit conservée dans la prochaine génération de la population évolutive. comme pour trouver la solution finale excellente établie à travers l'évolution étape par étape. Dans NSGA-II, la stratégie de tri rapide non dominé est utilisée pour évaluer le statut dominant de chaque individu dans la population évolutive, et selon le statut dominant de chaque individu, la population évolutive est divisée en différents sous-groupes, ce qui est un moyen nécessaire pour calculer l'aptitude de chaque individu dans la foule et d'évaluer la qualité de chaque individu. Dans la stratégie de tri rapide non dominé, la première étape de cette stratégie consiste à calculer le nombre de solutionss_i dominantes correspondantes de chaque individu (par exemple, un individu i) et le nombre d'individus n_i qui dominent l'individu dans la population évolutive. Lorsque le correspondant d'un individun_i est égal à 0, envoyez-le au premier niveau des individus non dominants.
Dans la stratégie de tri rapide non dominé, il déclare que lorsque le grade dominant des individus non dominants à un niveau est déjà déterminé, le niveau de tous les individus sous le contrôle d'autres individus n'est plus pris en compte. Compte tenu de cela, pour l'ensemble individuel dominant d'un individu s_i, chacun des individus (disons individu j) correspondant $n_j$dans l'ensemble individuel dominant de l'individu $s_i$devrait soustraire 1. Après la soustraction, quand $n_j = 0$, l'individu $j$sera placé dans le deuxième niveau non dominant . Le reste des individus de la population évolutive utilise la même manière pour trouver à quel niveau non dominant ils appartiennent. Une fois que chacun des individus de la population évolutive a trouvé son propre niveau non dominant, les individus correspondants du niveau le plus bas sont proches du front de Pareto réel du problème d'optimisation multi-objectifs à optimiser. Ainsi, les individus correspondants du niveau le plus bas sont plus supérieurs que les autres individus de la population évolutive de l'algorithme NSGA-II. Et l'algorithme NSGA-II utilise exactement une probabilité plus grande pour sélectionner le niveau le plus bas par rapport au niveau haut dans la prochaine génération de la population évolutive, de sorte que l'algorithme NSGA-II puisse évoluer vers la direction du jeu d'optimisation réel de Pareto du problème d'optimisation multiobjectifs pour être optimisé.

Crowded distance

Le but de l'algorithme d'optimisation multi-objectifs n'est pas seulement de trouver l'ensemble optimal de Pareto qui est proche de l'ensemble optimal de Pareto réel, mais aussi de trouver les solutions dans l'ensemble optimal de Pareto ayant une très bonne distribution, de sorte que mieux et plus raisonnable des solutions peuvent être fournies au décideur. Le but de l'algorithme d'optimisation multi-objectifs est de trouver une série de distribution uniforme de la solution à choisir par les décideurs. Afin de bien obtenir la distribution de l'ensemble de solutions, divers algorithmes évolutifs ont utilisé de nombreux types de mesures de traitement. La plupart des algorithmes évaluent la distribution de chaque individu en construisant une fonction qui considère la distribution de la distribution individuelle. Dans l'algorithme NSGA-II, en introduisant la fonction de distance d'encombrement, le degré de congestion de chaque individu et des autres individus est calculé, et l'agrégation locale de l'ensemble de solutions est évitée par une certaine stratégie. La fonction de distance d'encombrement de l'algorithme NSGA-II peut être exprimée comme suit :

$$\text{PCD(K)} : \frac{1}{n}\sum_{i=1}^{n}\frac{f_i(k+1)-f_i(k-1)}{f_i^{max}-f_i^{min}}$$

où $PCD(K)$ représente la valeur de la distance d'encombrement du $K-i^{me}$ individu, $f_i(k+1)$ représente la valeur de la fonction objectif du $K+1$-i^{me} individu, et f_i^{max} et f_i^{min} représentent séparément le i^{me} Valeur maximale et minimale de la valeur de la fonction objectif dans l'ensemble de solutions. L'expression mathématique de la fonction de distance d'encombrement peut être vu que le PCD est la valeur moyenne de la distance euclidienne et de chaque fonction objective de l'individu.

Lors du calcul de la valeur PCD de chaque individu, nous devons remarquer que la valeur de la fonction de distance d'encombrement correspondant au point de frontière de chaque fonction objectif est définie comme 1. À partir de l'expression mathématique PCD, nous pouvons voir que la valeur de la valeur PCD de chaque individu varie de 0 à 1. Le but de l'introduction du concept PCD dans l'algorithme NSGA-II est d'évaluer l'encombrement des uns et des autres dans le même niveau non dominé. Une valeur PCD plus élevée peut indiquer à l'algorithme NSGA-II que l'individu correspondant est plus rare avec les individus environnants. Les individus qui ont une petite valeur PCD ont une très bonne distribution dans la population évolutive dans l'algorithme NSGA-

II. Selon l'introduction de la fonction PCD dans l'algorithme NSGA-II, les individus du front de Pareto auront une grande distribution dans l'ensemble optimal de Pareto, de sorte qu'il puisse garantir l'uniformité et la diversité de la population évolutive dans l'algorithme NSGA-II.

Algorithme

Opérateur d'assignation de la $Crowded$ distance [27]

Données :

I :vecteur d'indice.

$\mathscr{F}_k$: Sous-ensemble de $pareto$

d_i : $Crowded$ distance de la solution i.

f_i : Fonction d'objectif.

f_m^{max} : Valeur maximale de la fonction d'objectif m.

f_m^{min} : Valeur minimale de la fonction d'objectif m.

Début

Initialiser $l = \|\mathscr{F}\|$

Pour chaque $z^{(i)} \in \mathscr{F}$ faire

Initialiser $d_i = 0$

Pour chaque fonction objectif f_j

Trouver la vecteur d'indice $I^j = trier(f_j, <)$

Pour chaque fonction objectif f_j

Assigner $d_1^m = d_l^m = \infty$

pour chaque z$^i i = 2.....(l-1)$ faire

Calculer

$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{I_{j+1}^m} - f_m^{I_{j-1}^m}}{f_m^{max} - f_m^{min}}$

Fin

Sélection environnementale basée sur PCD :

Dans l'algorithme d'évolution multiobjectifs, les individus évoluent vers l'ensemble de solutions optimal en fonction d'une certaine stratégie d'évolution. Dans l'algorithme NSGA-II, la stratégie

d'évolution de la population peut être conclue comme selon une stratégie de tri rapide non dominé pour déterminer le niveau non dominé de chaque individu. En comparant la valeur PCD et le niveau non dominé de chaque individu pour déterminer les individus d'élite de la population actuelle et de l'ancienne population et sauver les individus d'élite à la génération suivante, puis l'évolution des individus dominants dans l'algorithme NSGA-II est terminée. Dans l'algorithme NSGA-II, la sélection des individus à la génération suivante pour l'évolution suit les deux règles suivantes :

(1) si le niveau non dominant d'un individu p est supérieur à q, alors le premier individu est choisi pour la génération suivante ; (2) si le niveau non dominant d'un individu p est égal à q, dans cette situation, la valeur PCD de ces deux individus doit être comparée. Dans l'algorithme NSGA-II, celui avec une valeur de PCD plus petite sera sélectionné pour la population d'évolution de prochaine génération.

Dans l'algorithme NSGA, $p < q$ est défini comme la forme mathématique suivante pour exprimer la stratégie de sélection environnementale :

$$p < q \begin{cases} p_{rank} < q_{rank} \\ ou \\ p_{rank} < q_{rank}, PCD(p) > PCD(q) \end{cases} \tag{4.1}$$

Grâce à la stratégie de sélection d'environnement de l'algorithme NSGA-II, nous pouvons voir quand le niveau non dominant de deux individus est différent, dans cette situation l'individu avec un niveau non dominant plus petit, de sorte que l'algorithme NSGA-II peut assurer la population d'élite ; au contraire, lorsque le niveau de non-dominance de deux individus est le même, le choix de l'individu avec une valeur de distance d'encombrement plus élevée peut assurer la diversité de la population, de sorte que la vitesse d'évolution de l'algorithme NSGA-II peut être accélérée.

Étapes de calcul de l'algorithme NSGA-II

Étape 1 : Initialisez aléatoirement la population initiale et la valeur de chaque paramètre dans l'algorithme NSGA-II, puis définissez le numéro d'itération sur 1 ; puis lancez le calcul d'itération de l'algorithme

Étape 2 : Déterminez si le numéro d'itération atteint actuellement les temps d'itération les plus élevés définis à l'étape 1 ; si le premier est plus grand que le second, terminez le calcul de l'itération et sortez l'ensemble optimal de Pareto ; si le premier n'est pas plus grand que le second, continuez le calcul de l'itération jusqu'à ce que les deux nombres soient égaux l'un à l'autre.

Étape 3 : Utilisez les stratégies de tri rapide non dominé pour effectuer la population évolutive. Et puis effectuez l'opération génétique comme la sélection, le croisement et la mutation de la population pour former la nouvelle génération qui est appelée Q dans cet article.

Étape 4 : Combinez la population P et la population Q en tant que nouvelle population de mélange appelée R, et exécutez la stratégie de tri rapide non dominé pour chaque individu de la population R ; puis calculez la distance d'encombrement de chaque individu de la population R et triez-les.

Étape 5 : Conserver les individus d'élite comme la nouvelle génération de la population évolutive que nous appelons P_{t+1}dans notre article.

Étape 6 : Ajoutez 1 au numéro d'itération $i = i + 1$, et un calcul d'itération est effectué ; puis passez à l'étape 2 et continuez le reste du calcul de l'itération.

Algorithme NSGA-II [27]

Données :

D : Opérateur d'assignation de $Crowded$ distance.

$\mathscr{F}_k$:Sous-ensemble de Pareto.

M : opérateur de mutation.

ρ :population d'individus.

$\wp$:population des enfants.

$\mathscr{R}$:population enfants - individus.

S : Opérateur de sélection.

T :opérateur de tri par ensemble de Pareto.

X : opérateur de croisement.

z_k^i :solution i faisant partie de l'ensemble de Pareto k .

début

Créer une population initiale ρ_0 et $\wp_0$

pour chaque itération t faire

$\mathscr{R}_t = \rho_t \cup \wp_t$

Appliquer $\mathscr{R}^{'} = \mathrm{T}(\mathscr{R})$

Assigner $\rho_{t+1} = \emptyset$

Tant que $\|\rho_{t+1}\| < N$ faire

k=k+1 et $\rho_{t+1} = \rho_{t+1} \cup \mathscr{F}_k$

Calculer $D(z_k^{(i)}) \forall i \in \mathscr{F}_k$

Trier $(\mathscr{F}_k, <)$

i=1

Répéter

$\mathscr{F}_k = \mathscr{F}_k \, z_k^i$

i=i+1

Jusqu'à ce que $\|\rho_{t+1}\| = N$

Appliquer

$\wp_{i+1} = M(X(S(\rho_{(i+1)})))$

Fin

4.1.2 Stratégie d'amélioration

Bien que l'algorithme NSGA-II soit largement et avec succès utilisé dans de nombreux domaines comme un excellent algorithme, il a encore de nombreux inconvénients à régler. Ces dernières années, l'algorithme de recherche heuristique étant largement utilisé dans de nombreux aspects, de nombreux chercheurs ont découvert que de nombreux algorithmes évolutifs avaient un problème de capacité de recherche locale médiocre, ce qui est très important pour les performances de calcul de l'algorithme. L'algorithme avec une meilleure capacité de recherche locale peut utiliser un moyen plus précis pour rechercher la solution optimale du problème d'optimisation à régler.

Un meilleur individu peut être trouvé en recherchant autour d'un individu non dominé. Ainsi, la recherche autour des individus non dominés pour un meilleur individu est nécessaire. Les études montrent que la capacité de recherche locale de l'algorithme NSGA-II original est faible ; surtout lorsqu'elle est proche de la frontière de Pareto à la fin du processus évolutif, l'efficacité évolutive est fortement réduite. Pour améliorer la capacité de recherche locale, la stratégie de recherche locale est ajoutée à l'algorithme NSGA-II après la génération de la population évolutive suivante. La stratégie de recherche locale explore un individu de la population évolutive pour déterminer si une solution plus performante peut être trouvée. Il existe deux types de solution mentionnée avec de meilleures performances : (1) le nouvel individu domine l'original ; (2) l'individu nouveau et l'individu d'origine ne se dominent pas, mais la forme physique du nouvel individu est meilleure que celle d'origine. Donc, pour un individu x, recherchez un nouvel individu $x^{'}$dans son voisinage, si $x^{'}$ qui est un nouvel individu recherché dans le voisinage de l'individu x domine x ou si son aptitude correspondante est meilleure, remplacez ce dernier ; sinon procédez à la recherche locale.

Les étapes spécifiques de la stratégie de recherche locale sont les suivantes.

Étape 1 :définissez le temps d'itération de la recherche locale $i = 1$ et lancez le calcul de la recherche locale.

Étape 2 :pour chaque solution $x^{'}$de l'ensemble optimal de Pareto, générez une nouvelle solution réalisable x autour d'elle, et calculez son aptitude.

Étape 3 :si $x^{'}$domine x ou $fitness(x^{'}) < fitness(x)$, alors remplacez x par $x^{'}$, sinon, maintenez x inchangé.

Étape 4 : $i = i + 1$

Étape 5 : si $i = Non$, arrêtez le calcul de recherche locale et sortez l'ensemble optimal de Pareto vers l'algorithme NSGA-II ; sinon, continuez le calcul de la recherche locale.

4.2 Les travaux de simulation

4.2.1 Sélection des paramètres

Il existe de nombreux paramètres dans les algorithmes NSGA-II tels que la taille de la population, les temps d'itération, la probabilité de croisement et la probabilité de mutation. La valeur de ces paramètres a une grande influence sur les performances de calcul de l'algorithme NSGA-II. Ainsi, lorsque nous utilisons l'algorithme NSGA-II, la sélection de la valeur du paramètre devient un problème très important. Pour certains problèmes multiobjectifs compliqués, une petite taille de population ne peut pas aider l'algorithme NSGA-II à atteindre un ensemble optimal de Pareto très idéal. Une très grande taille de population peut ralentir la vitesse de calcul de l'algorithme NSGA-II , bien qu'elle puisse obtenir un bon ensemble optimal de Pareto. Donc, comment sélectionner la valeur de la taille de la population en fonction de la complexité du problème multiobjectifs est un problème à approfondir. À l'heure actuelle, la valeur de la taille de la population est sélectionnée par l'expérience des utilisateurs. Le chercheur ajuste la valeur de la population étape par étape, jusqu'à ce que l'ensemble optimal de Pareto soit stable. L'algorithme NSGA-II est une variante de l'algorithme GA ; ils ont beaucoup de choses en commun bien que le premier soit un algorithme multi-objectif tandis que le second est un algorithme à objectif unique. Ainsi, le NSGA-II utilise également une méthode itérative pour calculer le problème proposé.
Ensuite, un paramètre très important doit être évalué en premier, à savoir les temps d'itération. Un temps d'itération très long peut aider l'algorithme NSGA-II à obtenir un très bon ensemble optimal de Pareto, mais il peut également augmenter le temps de calcul, ce qui est mauvais pour les utilisateurs de l'algorithme. Bien que de petits temps d'itération ne puissent garantir que l'algorithme NSGA-II atteigne un ensemble optimal de Pareto idéal, il peut aider l'algorithme NSGA-II à calculer très rapidement. De nombreux chercheurs utilisent désormais une méthode étape par étape pour ajuster la valeur du temps d'itération pour la sélectionner. Si l'ensemble optimal de Pareto ne change pas beaucoup dans l'algorithme NSGA-II , la valeur d'itération peut être utilisée comme valeur finale de ce paramètre de l'algorithme NSGA-II . La manipulation génétique comprend des opérations de sélection, de croisement et de mutation , ce qui est très important dans l'algorithme NSGA-II . Dans l'algorithme NSGA-II , les opérations génétiques sont le principal

moyen d'évolution qui détermine directement les performances des résultats. Dans les opérations génétiques, de nombreux paramètres peuvent modifier les performances d'évolution génétique de l'algorithme NSGA-II . Les paramètres génétiques sont donc également très importants et sont essentiels aux performances de calcul de l'algorithme NSGA-II . Dans l'algorithme NSGA-II , la sélection du tournoi est utilisée dans le processus de sélection, dont le paramètre est appelé valeur du tournoi. La valeur du tournoi détermine la pression de sélection ; plus la valeur est élevée, plus la pression de sélection est grande. Dans l'algorithme NSGA-II , la valeur du tournoi est suggérée à 2. Le processus de croisement de l'algorithme NSGA-II utilise une sorte de méthode de croisement binaire simulé pour croiser deux individus.

Il existe également un paramètre dans le processus de croisement qui est appelé paramètre de distribution de croisement. La valeur du paramètre de distribution croisée détermine la probabilité de la probabilité de la progéniture et du père de l'individu. Plus le paramètre de distribution croisée est grand, plus la probabilité de la progéniture et du père est grande. Cet article définit la valeur du paramètre de distribution croisée sur 20. Dans l'algorithme NSGA-II , une sorte de mutation polynomiale est utilisée dans le processus de mutation, dont le paramètre à l'intérieur est appelé paramètre de distribution de mutation. Le paramètre de distribution de mutation est plus ou moins le même que le paramètre de distribution de croisement. Dans notre article, la valeur du paramètre de distribution de mutation est fixée à 20. Comme la stratégie de recherche locale est ajoutée dans l'algorithme NSGA-II , le temps d'itération de recherche locale doit être évalué. Dans cet article, la valeur du temps d'itération de recherche locale est définie sur 10.

4.2.2 Simulation

Afin de rechercher l'effet d'amélioration de NSGA-II , cet article sélectionne un problème multi-objectif classique pour la simulation. Le premier problème est appelé KUR qui est le problème MISA. La formulation mathématique est la suivante :

$$f_1 = -10\exp(-0.2 + sqrt(x^2 + y^2)) - 10\exp(-0.2 + sqrt(y^2 + z^2))$$

$f_2 = |x|^{0.8} + 5 * \sin(x^3) + |y|^{0.8} + 5 * \sin(y^3) + |z|^{0.8} + 5 * \sin(z^3)$
où x, y et z sont les variables de contrôle et leurs domaines sont tous [5, -5]

La figure 1 montre les fronts Pareto du NSGA-II et du I-NSGA-II de ce problème. À partir de la figure 1, nous pouvons voir la différence des deux fronts de Pareto d'algorithmes : la distribution des fronts de Pareto de I-NSGA-II est plus uniforme que l'algorithme NSGA-II. De plus, les fronts Pareto de l'I-NSGA-II sont plus proches de l'origine des coordonnées. Ainsi, l'I-NSGA-II peut obtenir une meilleure solution optimale Pareto que l'original.

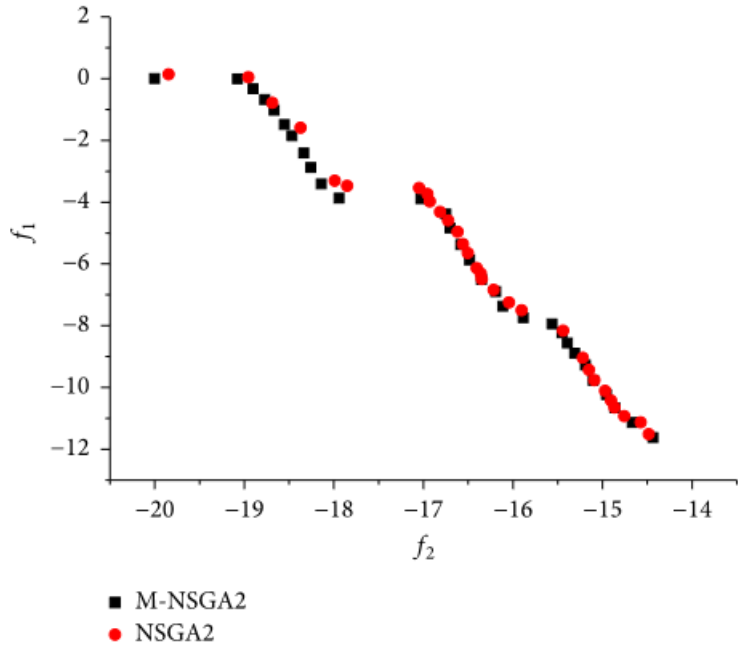


FIGURE 4.1 – Deux fronts Pareto du problème de KUR .

Pour approfondir et expliquer la différence entre les deux résultats, nous pouvons conclure que, simplement en raison de la stratégie de recherche locale, lorsque l'algorithme NSGA-II trouve une solution d'ensemble de Pareto dans chaque calcul itératif, la recherche locale peut commencer une recherche plus approfondie et plus précise de meilleure solution, et c'est pourquoi l'ensemble de Pareto du M-NSGA-II est plus proche du point d'origine des coordonnées que l'algorithme d'origine.

Le deuxième cas de simulation est également un problème multi-objectif classique appelé pro-

blème ZDT3. Le problème ZDT3 peut être exprimé mathématiquement comme suit :

$$minf_1(x) = x_1$$
$$minf_2(x) = g[1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}sin(10\pi f_1)]$$
$$g = 1 + 9\sum_{i=1}^{i=k} \frac{x_i}{n-1}$$

où $0 < x_i < 1$, et n = 30

La comparaison des deux algorithmes NSGA-II différents dans les problèmes ZDT3 est illustrée à la figure 2. À partir de la figure 2, nous pouvons clairement voir la différence des deux fronts de Pareto.

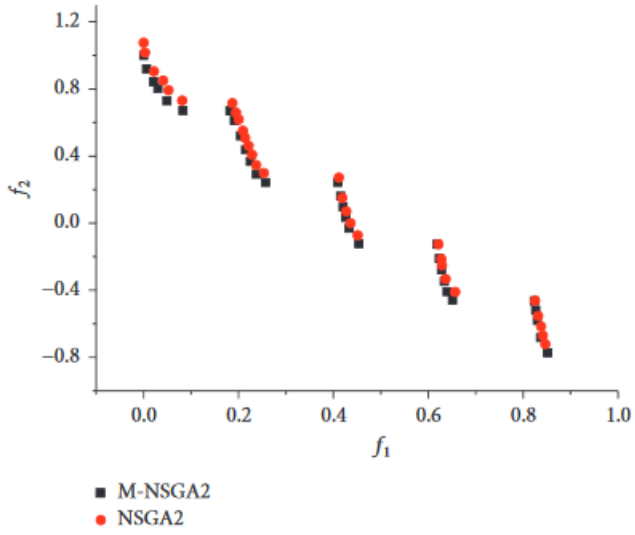


FIGURE 4.2 – Deux fronts Pareto du problème de ZDT3.

Le front de Pareto de l'algorithme M-NSGA-II peut conclure le second, ce qui signifie que le premier front de Pareto est plus proche du front de Pareto réel du problème ZDT3. En comparant les résultats de ce cas avec le premier cas, nous pouvons également voir que plus le problème est complexe, plus la supériorité du M-NSGA-II sera évidente. En voyant cette comparaison, nous pouvons conclure que la stratégie modifiée dans l'algorithme M-NSGA-II peut améliorer l'effet d'optimisation de l'algorithme NSGA-II.

Conclusion générale

Les problèmes d'optimisation multi-objectifs sont de plus en plus étudiés car ils ont un grand intérêt dans l'industrie, où plusieurs objectifs concurrents doivent être satisfaits.Ils sont la plupart du temps NP-difficiles même si leur version mono-objectif ne l'est pas.Contrairement aux problèmes d'optimisation mono-objectif, il existe généralement plu-sieurs solutions Pareto optimales ou efficaces en optimisation multi-objectifs, c'est-à-dire des solutions qu'on ne peut améliorer sur un objectif sans les détériorer sur un autre.Comme il n'existe aucune solution meilleure en tout point qu'une autre, un compromis différent selon les personnes doit être choisi. Le choix est donc subjectif, et il est indispensable de proposer l'ensemble des choix possibles afin de ne pas exclure une possibilité.L'optimisation multi-objectifs est donc avant tout un outil d'aide à la décision, et c'est une personne qui prendra la décision finale.

Nous avons présenté dans ce mémoire les travaux de recherche sur les problèmes d'optimisation multi-objectif par la méthode NSGA-II (non dominated sorting genetic algorithm).

Dans notre étude, nous nous sommes particulièrement intéressés aux problèmes d'optimisation multi-objectif pour lesquels relativement peu de travaux ont été réalisés. Le noyau du thème de notre mémoire est principalement proposer une simple stratégie de gestion des contraintes avec NSGA-II qui convient bien à tous les algorithmes évolutifs multiobjective et le travail présenté s'articule autour de deux voltes :

• Le premier volte consiste à rappeler les notion d'optimisation multi-objectifs.

• Le deuxième volte consiste à applique la méthode NSGA-II pour résourde les problèmes d'optimisation multi-objectif.

Test sur la méthode NSGA-II(non dominated sorting genetic algorithm).

Exemple 1 :Le problème de Shaffer

$$(P): \begin{cases} minf_1(x) = x^2 \\ minf_2(x) = (x-2)^2 \\ x \in [-10^3, 10^3] \end{cases} \tag{4.2}$$

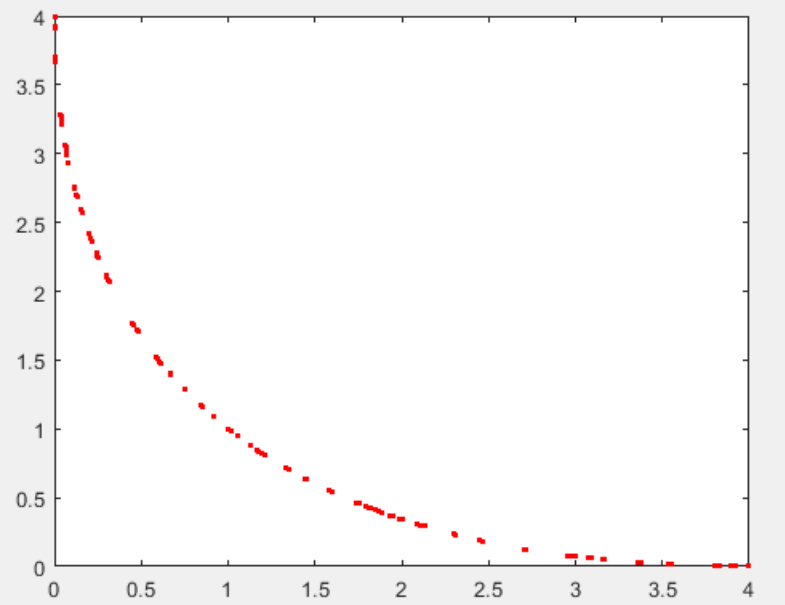


FIGURE 4.3 – .Nondominated solutions with NSGA-II on SCH

Exemple 2 :Le problème de Kursawe

$$(P): \begin{cases} minf_1(x) = \sum_{i=1}^{n-1}(-10\exp(-0.2\sqrt{x_i^2 + x_{i+1}^2})) \\ minf_2(x) = \sum_{i=1}^{n}((|x_i|)^{0.8} + 5\sin x_i^3) \end{cases} \tag{4.3}$$

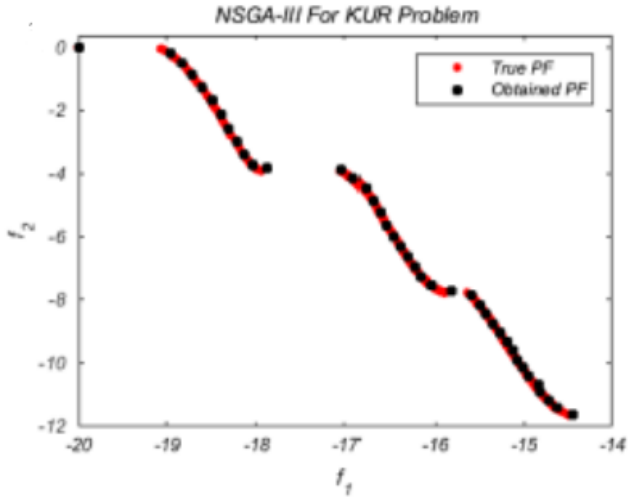


FIGURE 4.4 – .Nondominated solutions with NSGA-II on KUR

Exemple 3 :Le problème de ZDT1

$$(P): \begin{cases} minf_1(x) = x_1 \\ minf_2(x) = g(x)[1-\sqrt{x_1/g(x)}] \\ g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i = 0 \quad i = 2,...,n \\ x_1 \in [0,1] \end{cases} \tag{4.4}$$

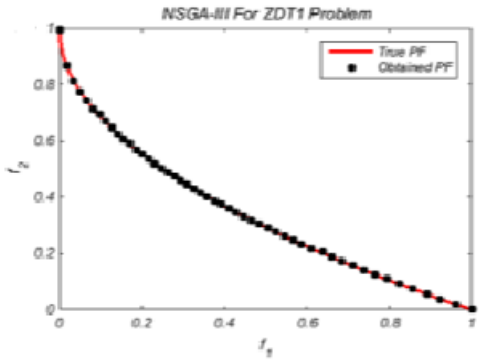


FIGURE 4.5 – .Nondominated solutions with NSGA-II on ZDT1

Exemple 4 :Le problème de ZDT2

$$(P): \begin{cases} minf_1(x) = x_1 \\ minf_2(x) = g(x)[1-(x_1/g(x))^2] \\ g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i = 0 \quad i = 2,...,n \\ x_1 \in [0,1] \end{cases} \tag{4.5}$$

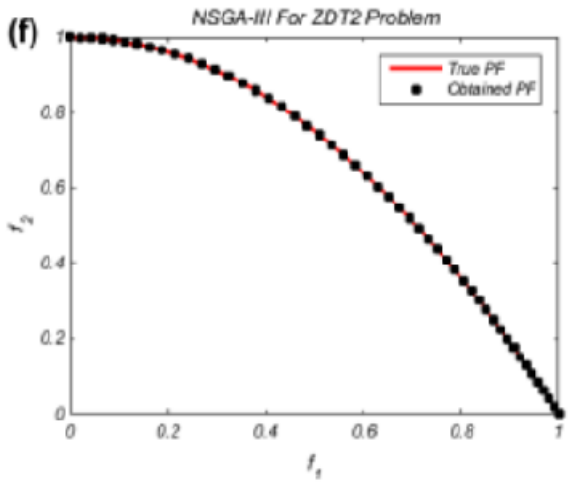


FIGURE 4.6 – .Nondominated solutions with NSGA-II on ZDT2

Exemple 5 :Le problème de ZDT3

$$(P): \begin{cases} minf_1(x) = x_1 \\ minf_2(x) = g(x)[1-\sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)] \\ g(x) = 1+9(\sum_{i=2}^{n} x_i)/(n-1) \\ x_i = 0 \quad i = 2,.....,n \\ x_1 \in [0,1] \end{cases} \tag{4.6}$$

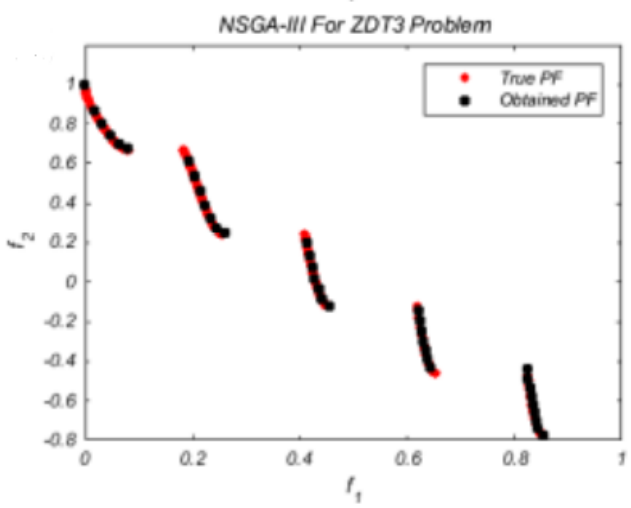


FIGURE 4.7 – .Nondominated solutions with NSGA-II on ZDT3

Exemple 6 :Le problème de ZDT4

$$(P): \begin{cases} minf_1(x) = x_1 \\ minf_2(x) = g(x)[1 - \sqrt{x_1/g(x)}] \\ g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}[x_i^2 - 10\cos(4\pi x_i)] \\ x_i = 0 \quad i = 2, ..., n \\ x_1 \in [0,1] \end{cases} \tag{4.7}$$

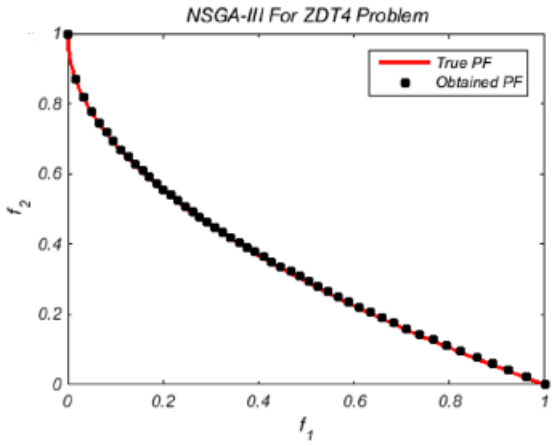


FIGURE 4.8 – .Nondominated solutions with NSGA-II on ZDT4

Exemple 7 :Le problème de ZDT6

$$(P): \begin{cases} minf_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1) \\ minf_2(x) = g(x)[1-(f_1(x)/g(x))^2] \\ g(x) = 1 + 9[(\sum_{i=2}^{n} x_i)/(n-1)]^{0.25} \\ x_i = 0 \quad i = 2, ..., n \\ x_1 \in [0,1] \end{cases} \tag{4.8}$$

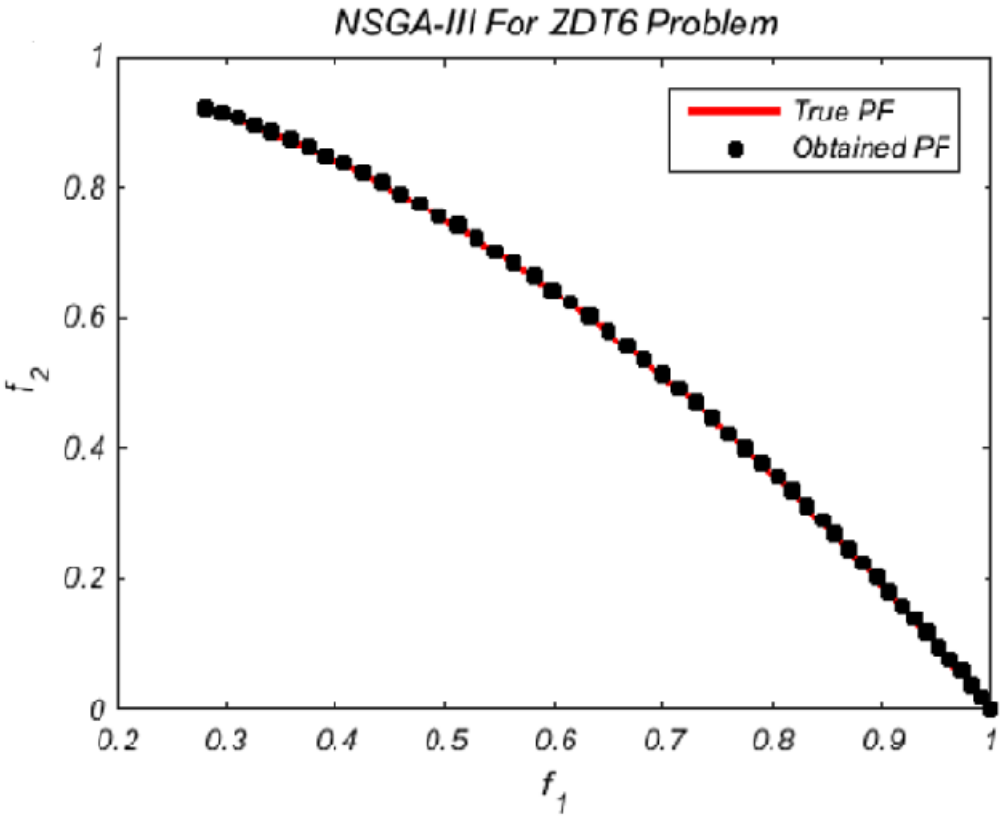


FIGURE 4.9 – .Nondominated solutions with NSGA-II on ZDT6

Bibliographie

[1] V.L.Huang A.K.Qin and P.N.Suganthan. "differential evolution algorithm with strategy adaptation for global numerical optimization,". *IEEE Transactions on Evolutionary Computation,*, vol.19,no. 2, :pp.398–417„ 2009.

[2] Johan Andersson. *Multiobjective optimization in Engineering Design,Applications to Fluid Power System (thèse de Doctorat)*. PhD thesis, Université de Linköping, Sweden ;, 2001.

[3] Alain Berro. *Optimisation Multiobjectif et stratégies d'evaluation en environnement dynamique*. PhD thesis, University de Toulouse, 18 december 2001.

[4] B.Milosevic and M.Begovic. "nondominated sorting genetic algorithm for optimal phasor measurement placement ,". *IEEE Transaction on Power systems*, vol.18.no.1 :pp. 69–75, 2003.

[5] G. Lucian C. Razvan and M. Ioan. "an oop matlab extensible framework for the implementation of genetic algorithms. part i : the framework,". *in Proceedings of the 8th International Conference Interdisciplinarity in Engineering (INTER-ENG '14)*, vol.19 : pp. 193–200, 2015.

[6] M.Caldora Costa. *Optimisation de Diapositifs Electromagnétiques dans un Contexte d'analyse par la Méthode des Eléments Finis*. PhD thesis, L.E.GLaboratoire d'électrotechnique de Grenoble, 17 juillet 2001.

[7] D.A.Van.Veldhuizen. “multiobjective evolutionary algorithms :classification analyses and new innovation ,ph.d,”. *Graduate school of Engineering ;Air Force Institute of technology ;Wright Ptterson AFB,Ohio,USA,janvier*, 1999.

[8] D.E.Goldberg. “genetic algorithms in search ,optimisation and machine learning ,”. *Addison Wesley pubishing compagny*, 1989.

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

[10] D.E.Goldberg and J.Richardson. “genetic algorithms with sharing for multimodal function optimisation and their application ,”. *proceedings of the second International conference on genitic algorithms*, vol. 37, no. 4 :pp.41–49, 1987.

[11] D.E.Goldberg and K.Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms*, pages pp.69–93. 1991.

[12] D.Karaboga and B.Basturk. “a powerful and efficient algorithmfor numerical function optimization : artificial bee colony (abc) algorithm,”. *Journal of Global Optimization*, vol. 39, no. 3 :pp.459–471,, 2007.

[13] E.Bonomi and J.L.Lutton. “le recuit simulé pour la science ,”. pages pp.68–77, juillet1988.

[14] E.E.Korkmaz. “multi-objective genetic algorithms for grouping problems,”. *Applied*, vol. 33, no. 2 :pp. 179–192, 2010.

[15] E.Talbi. “métaheuristiques pour l’optimisation combinatoire multi-objectif : Etat de l’art ”. *Lille :LIFL*, 2001.

[16] M.laumanns E.Zitzler and L.thiele. “spea2.improving the strenght pareto evolutionary algorithm ,”. vol.3242,no.103 :pp .95–100, 2001.

[17] F.Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, vol.13 :pp.533–549, 1986.

[18] F.Glover and M.Laguna. Tabu search. Kluwer Academic Publishers Norwell MA, second edition, 1997.

[19] Goldberg. Genetic algorithms in search,optimistion and machine learning. publishing company, 1989.

[20] J.Koski H.Eschauer and A.Osyczka. Multicriteria design optimization :procedures and applicatins. Springer-Varleg, 1990.

[21] J.D.schaffer. "multiple objectif optimisation with vector evaluted genetic algorithms,". *PhD thesis,vanderbilt University*, 1984.

[22] J.D.Schoffer. *Multiple objective optimisation with vector evaluted genetic algorithms*. PhD thesis, venderbilt university, 1984.

[23] Baker J.E. Adaptive selection methods for genetic algorithms. In *Proceedings of an International Conference on Genetic Algorithms and their applications*, pages pp.101–111. July 1985.

[24] J.E.Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of an International Conference on Genetic Algorithms and their applications*, pages pp.14–21. July 1987.

[25] J.Holland. *Adaptation in artificial and natural systems*. PhD thesis, the University of Michigan Press, 1975.

[26] A.Pratap k.Deb and al. "a fast elitist non-dominated sortind genetic algorithm for multi-objectif optimisation : Nsga-ii,". *in parallel Problem Solving from Nature PPSN :6th International Confernce Paris ,Paris ,France ,Septembre 18-20,2000 Proceding*, vol.1917 of Lecture Notes in Computer Science :pp.849–858, 2002.

[27] A.Pratap K.Deb, S.Agrawal and T.Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization : Nsga-ii, proceedings of the parallel problem solving from nature vi conference. pages pp.849–858., 2000.

[28] S.Agarwal k.Deb, A.Pratap and Meyarivan. "a fast elitist multiobjectif genetic algorithem : Nsga-ii,". *IEEE Transaction on Evolutionary computation*, 6,no.2 :pp.182–197, 2002.

[29] J. Kennedy. “particle swarm optimization,”. *in Encyclopedia of Machine Learning*, pages pp.760–766, 2011.

[30] K.S.Tang K.F Man and S.Kwong. Genetic algorithms : concepts and applications [in engineering design]. In *Industrial Electronics*, pages pp.519–534. IEEE on, 43(5), 1996.

[31] K.Miettinen. Non linear multiobjective optimisation. Kluwer,Boston, 1999.

[32] Fuzzy Sets L.A.Zadeh. “information and control,”. vol. 8, :pp.338–353, 1965.

[33] T.G.Wang L.Wang and Y.Luo. “improved non-dominated sorting gentic algorithm (nsga)-ii in multi-objective optimization studies of wind turbine blades,”. *Applied Mathematics and Mechanics*, vol.32,no.6 :pp. 739–748, 2011.

[34] E.E.El-Araby M.Eghbal, N.Yrino and Y.Zoka. “optimal var expansion considering atc using strength pareto evolutionary algorithm,”. *in proceeding of the 40th North American Power Symposium (NAPS’08)*, pages pp.550–555, 2008.

[35] K.Lansey M.Eusuff and F.Pasha. Shuffled frog-leaping algorithm : a memetic meta-heuristic for discrete optimization,”. *Engineering Optimization*, vol. 38, no. 2 : pp.129–154, 2006.

[36] M.kaya. “autonomous classifiers with understandable rule using multi-objective genetic algorithms,”. *Expert Systems with Applications*, vol. 37, no. 4 :pp. 3489–3494, 2010.

[37] M.Miki M.Kim, T.Hiroyasu and al. “spea2+ :improving the performance of the strength pareto evolutionary algorithm2,”. *in Parallel Problem solving from nature –PPSN VIII : 8th International conference ,Birmingham ,UK,Septrmbre 18–22*, vol.3242 :pp. 742–751, 2004.

[38] M.Okeeffe and M.Q.Ginnéide. “a stochastic approach to automated design improvement,”. 2003.

[39] M.P.Fourman. “in procedings of the first international confernce on genetic algorithms (icga) ,”. *PhD thesis,vanderbilt University*, 1985.

[40] M.Sakawa and H.Yano. “an interactive fuzzy satisficing method formultiobective linear programming problems with fuzzy parameters,”. *fuzzy sets and systems*, vol. 28, :pp. 129–144, 1988.

[41] M.Salami and T.Hendtlass. “a fitness estimation strategy for genetic algorithms,”. *Lecture Notes in Computer Science*, vol.2358 :pp. 502–513, 2002.

[42] N.Benlahrache. Optimisation multi-objectif pour l'alignement multiple de séquences, 2007.

[43] M.N.Rosenbluth N.Metropolis and H.A.Teller. Equation of state calculation by fast computing machines. *Journal of Chemical Physics*, vol.21(6) :10871092,, 1953.

[44] N.Srinivas and K.Deb. Multiobjective function optimization using nondominated sorting genetic algorithms. *Evolutionary Computation Journal*, pages pp. 221–248., 1995.

[45] S.Krichen O.Abdelkhalek and A.Guitouni. “a genetic algorithm based decision support system for the multi-objective node placement problem in next wireless generation network,”. *Expert SystemswithApplications*, vol.33, no.8 :pp. 278–291,, 2015.

[46] P.Hansen. The steepest ascent mildest descent heuristic for combinatorial programming. In *présenté au Congress on Numerical Methods in Combinatorial Optimization*, 1986.

[47] S.Hammadi R.Benagad. Optimisation des coûts carburants air algerie, 2008.

[48] R.Berro and P.V.Hentenrych. A two-stage hybride search for the vehicle routing problem with time windows. *transportation science*, vol.38(4) :pp.515–530, Nouvember 2004.

[49] H.D. Sherali S. Bazarra and C.M. Shetty. Nonlinear programming, theory and algorithms. Second edition, 1993.

[50] C.D.Gelatt S.Kirkpatrik and K.P.Vecchi. Optimization by simulated annealingp. *science,*, vol.220 :pp. 671—-680, 1983.

[51] V.Cerný. A thermodynamical approach to the traveling salesman problem : an efficient simulation algorithm. *Journal of Optimization Theory and Applications*, pages pp.41—-51, 1985.

[52] V.Chankong and Y.Y.Haimes. "multiobjective decision making :theory and methodology. *North series in system and science and Engeneering*, page pp. 406, 1983.

[53] Patrick Siarry Yann Collette. *Optimisation Multiobjectif*. Edition eyrollrs,75240 Paris Cedex 05,France.

[54] J.S.Yoo Y.B.Park and H.S.Park. "a genetic algorithm for the vendor-managed inventory routing problem with lost sales," expert systemswithapplications,". *Expert SystemswithApplications*, vol. 53, no. 1 :pp. 149.–159, 2016.

[55] Y.Y.Haimes. *Modeling and Control of the Pollution of Water Resources Systems via Multilevel Approch*. PhD thesis, New -York, 1971.

[56] L.S.Lasdon Y.Y.Haimes and D.A.Wismer. Formilation of the problèmes of integrated system identification and system optimisation, 1986.

[57] W.A.Hall Y.Y.Haimes and H.T.Freedman. Multiobjevtive optimisation in water resources systems. editions Elsevier scientific publication company, 1975.