République Algerienne Démocratique et populaire

Ministère de l'ensignement supérieur et de la recherche scientifique

Université Saad Dahlab de Blida1

Faculté des Sciences

Département de Mathématiques

Mémoire de fin d'études en vue de l'obtention du diplôme

de Master en Mathématiques

Option :

Modélisation Stochastique et Statistique

Thème

Un Critère d'Arrêt pour les MOEA basé sur la théorie de l'Entropie

Présenté par :

- Hameche Houssem Eddine
- Mohammed Cherif Hassen

Devant le jury :

M.Président : O.TAMI

M.Examinateur :H.El MOSSAOUI

M.Promoteur : M.AIT AKKACHE

Année Universaire : 2016/2017

ملخص

الهدف الأساسي من التحسين المتعدد الأهداف المعتمد على هيمنة باريتو هو ايجاد مجموعة من الحلول الغير مهيمنة والتي تحقق مختلف التسويات الممكنة بين الأهداف المحسنة.

نقدم في هدا المشروع وقف المعيار من أجل حل أمثل لمشاكل مع أكثر من هدفين. مع الحفاظ على توزيع واسع موحد بين الحلول.

. RÉSUMÉ

L'optimisation multi-objective à base de dominance de Pareto a pour objectif essentiel de fournir un ensemble de solutions non-dominées réalisant divers compromis entre les objectifs à optimiser.

Nous presentons dons ce projet le critère d'arr êt dams les MOEA pour résoudre les problemes d'optimisation avec plus de deux objectifs.

Abstract

Multi-objective optimization based on Pareto dominance aim sat providing a set of non-dominated solutions that achieve various compromises between the objective functions that need to be optimized.

We present in this project, stop criterion from MOEA, to solve optimization problems with more than two objectives.

Remerciements

Ce mémoire à été réalisé dans le cadre du projet de fin d'études de l'année 2017 au département de mathématique.

On adresse en premier lieu notre reconnaissance à notre DIEU tout puissant, de nous avoir permis d'en arriver là, d'aller jusqu'au bout du rêve car sans lui rien n'est possible.

Nous tenons à présenter nos vifs remerciements à notre encadreur Dr.Ait Akkache pour son encadrement et ses conseils qui ont été précieux et bénéfiques, de nous avoir fait bénéficier de ses compétences, ses qualités humaines et de sa disponibilité pour la réalisation de notre projet de fin d'études.

Nous tenons aussi à exprimer nos plus profonds sentiments de reconnaissance à nos chers professeurs qui n'ont cessé de nous aider et de nous encourager pendant les cinq ans de notre formation au sein du département de mathématiques.

On n'oublie pas de dire un grand merci à toutes les personnes, tous les professionnels qui ont contribuées de près et de loin à l'enrichissement de notre travail et à notre épanouissement intellectuel.

Nous remercions cordialement le jury qui a accepté d'examiner et d'évaluer notre travail.

Merci à tous

Dédicaces

J'ai le plaisir d'offrir ce modeste travail à tous ceux qui de près ou de loin ont aidé à sa réalisation.

A la mémoire de ma mère…

A mon père ;

A mes frères et mes sœurs chacun par son nom ;

A toutes mes amies ;

A tous ceux qui m'aiment ;

A tous ceux que j'aime

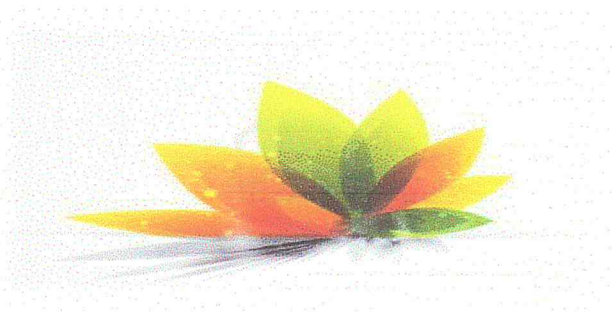


Table des matières

Table des figures

Liste des tableaux

LISTE DES ALGORITHMES

algorithmList of Algorithms

Introduction générale

Les ingénieurs, les économistes, les scientifiques et les décideurs se heurtent quotidiennement à des problèmes d'optimisation dans divers domaines tels que la conception de systèmes mécaniques, le traitement d'images, l'électronique ou la recherche opérationnelle pour n'en citer que peu. Un problème d'optimisation est défini par un ensemble de variables, une fonction objective et un ensemble de contraintes. L'espace de recherche est l'ensemble des solutions possibles du problème. Résoudre un problème d'optimisation consiste à trouver la ou les meilleures solutions (en minimisant et/ou maximisant la/les fonctions objectifs du problème posé), tout en satisfaisant un ensemble de contraintes définies par l'utilisateur. Dans la plupart des problèmes du monde réel, il ne s'agit pas d'optimiser seulement un seul critère mais plutôt d'optimiser simultanément plusieurs critères et qui sont généralement conflictuels. L'optimisation multi objectif consiste donc à optimiser simultanément plusieurs fonctions en cherchant le meilleur compromis possible sachant que l'amélioration d'un objectif entraine la détérioration d'un autre objectif. La notion de solution optimale unique dans l'optimisation mono objectif disparait pour les problèmes d'optimisation multi-objectif (MOP) au profit de la notion d'ensemble de solutions Pareto optimales.

Nous distinguons deux grandes classes de méthode dans la résolution des MOP : les méthodes déterministes (exactes) et les méthodes stochastiques (méta-heuristiques). Parmi les méta-heuristiques, on trouve la famille des algorithmes évolutionnaires qui ont connus un grand succès dans l'optimisation multi-objectifs.

Les algorithmes évolutionnaires (AE) sont inspirés de la théorie de l'évolution de Darwin. Ce sont des techniques d'optimisation itérative et stochastique, car ils utilisent itérativement des processus aléatoires. Ils font évoluer un ensemble de solutions (dite population) d'un problème donné en utilisant une fonction d'évaluation (fitness) afin de mesurer ces valeurs, dans l'optique de trouver les meilleurs résultats. Cette particularité donne aux AEs la capacité de trouver plusieurs solutions Pareto optimales en une seule exécution.

Lors de l'utilisation des algorithmes évolutionnaires multi-objectif (MOEA) pour les problèmes multi-objectifs continus, le nombre de génerations nécessaires pour assurer une bonne approximation du front de Pareto est une question critique qui incombe praticiens. Devant l'absence d'un critére adéquat, la pratique courante consiste á fixé a priori le nombre d'itérations ou á exécuter l'algorithme jusqu'á épuisement du temps disponible. Cependant, cette approche est à double tranchant, dans le sens, qu'un MOEA peut fournir des solutions sous-optimales, si le nombre de générations a priori fixé est insuffisant ou le temps de calcul disponible est court. En revanche, un grand nombre de générations , ou un temps de calcul trop long peut entraîner un gaspillage de ressources. De plus, ces critéres sont dépourvus de toute intelligence intrinséque comme par exemple, une évaluation intermittente ou finale de la qualité de l'approximation obtenue du PF par rapport au véritable front.

Compte tenu de cela, notre travail consiste en la présentation d'un nouveau test d'arrêt qui tient compte de la qualité intrinsèque des solutions trouvées. Ce critère d'arrêt est construit sur la base du concept de l'entropie de shanon , utilisée habituellement en théorie de l'information, et qui permet une rationalisation des moyens de calcul.[Notre critère incorporer dans un MOEA et une serie de simulations numériques sera effectuée pour une éventuelle validation].

Ce mémoire est organisé en quatre chapitres :
Le premier chapitre : est consacré aux principes de base de l'optimisation multi-objectif. On donnera aussi les phases de résolution d'un problème d'optimisation multi-objectif (MOP) et on finira par quelques méthodes de résolution exacte de MOP. Dans **le deuxième chapitre :** nous nous intéresserons de manière approfondie à l'évolution artificielle, nous présentons les notions nécessaires a la compréhension des EAs et leur application au cas de l'optimisation multi-objectif. **Le troisième chapitre :** constitue le noyau de ce mémoire et q'est composé d'un rappel sur la cette théorie de l'entropie suivie par une construction détaillée de notre test d'arrêt. Dans construction, on utilise des techniques de simulation statistique et des ruses pour gérer le flux des données.

Le dernier chapitre est consacré aux expérimentations effectuées sur des problèmes tests et qui permettent d'évaluer notre réalisation. Enfin, nous terminons par **une conclusion générale** qui sera une synthèse de notre travail et qui ouvre de nouvelles perspectives.

CHAPITRE 1

OPTIMISATION MULTI-OBJECTIF ET OPTIMALITÉ DE PARETO

1.1 Introduction

La plupart des problémes du monde réel nécessitent d'optimiser simultanément plusieurs critères et qui sont généralement conflictuels. Ce type d'optimisation est connu sous l'appellation d'optimisation multi-objective. Dans cette partie, nous présentons tout d'abord un ensemble de définitions lieés à l'optimisation multi-objective. Ensuite nous présentons quelques méthodes traditionnelles utilisées dans la résolution des problèmes d'optimisation multi-objectif.

1.2 Problème général de l'optimisation multi-objectif

1.2.1 Définition

Un problème d'optimisation [7] est la recherche du minimum ou du maximum d'une fonction . On peut aussi trouver des problèmes d'optimisation pour les quels les variables de la fonction à optimiser sont contraintes à évoluer dans une certaine partie de l'espace de recherche. Dans ce cas , on a une forme particulière de ce que l'on appelle un problème d'optimisation sous contraintes.

Mathématiquement parlant, un problème d'optimisation se présente sous la forme suivante :

$$(MOP) = \begin{cases} optimiser(f_1(\overrightarrow{x}),,...,f_M(\overrightarrow{x})) \\ avec\ g_i(\overrightarrow{x}) \geq 0,\ j = 1,2,...J \\ h_R(\overrightarrow{x}) = 0,\ R = 1,2,...R; \end{cases} \tag{1.1}$$

telle que :

• $\vec{x}$ est le vecteur de n variables de décision $\vec{x} = (x_1, x_2, ..., x_n)^T$.

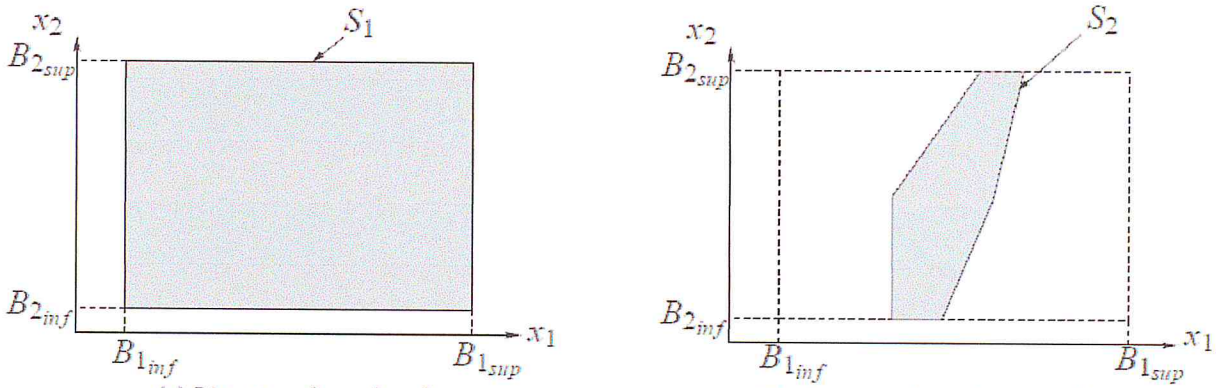
• M est le nombre de fonctions objectifs et $\vec{F}(\vec{x}) = ((f_1(\vec{x}), f_2(\vec{x}), ..., f_M(\vec{x}))^t$ est le vecteur des fonctions objectives .

• $g_i(\vec{x})$ *et* $h_k(\vec{x})$ représentent respectivement les contraintes d'égalité et d'inégalité. Ces contraintes délimite un espace de recherche restreint.

• en général on trouve deux types de contraintes d'inégalités.

1) Des contraintes du type $B_{i_{inf}} \leq x_i \leq B_{i_{sup}}$ dans ce cas l'espace de recherche est semblable à celui de la la figure 1-1-a(n=2).

2) Des contraintes de type $C(\vec{x}) \leq 0$ ou $C(\vec{x}) \geq 0$ et dans ce cas cet espace est du genre représenté dans la figure 1.1-b(n=2).



(a) L'espace de recherche (b) L'espace des valeurs réalisables

Figure 1.1 : Les différents types d'espaces derecherche.

1.3 Vocabulaire et définitions

• Fonction objectif :

C'est le non donné à la fonction $\vec{F}$ (on l'appelle encore **fonction de coût** ou **critère d'optimisation**). C'est cette fonction que l'algorithme d'optimisation va devoir "optimiser" (trouver un optimum).

• Variables de décision :

Elles sont regroupées dans le vecteur $\vec{x}$. C'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction $\vec{F}$.

• Minimum global :

Un "point" $\vec{x}^*$ est un minimum global de la fonction f, (cas mon-objectif, $\vec{F} = f$), si on a :
$f(\vec{x}^*) < f(\vec{x})$ quel que soit $\vec{x}$ telle que $\vec{x}^* \neq \vec{x}$.

• Minimum local fort :

Un "point" $\vec{x}^*$ est un minimum local fort de la fonction f si on a :
$f(\vec{x}^*) < f(\vec{x})$ quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$, où $V(\vec{x}^*)$ définit un "voisinage" de $\vec{x}^*$.

• Minimum local faible :

Un "point" $\vec{x}^*$ est un minimum local faible de la fonction f si on a :
$f(\vec{x}^*) \leq f(\vec{x})$ quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$, où $V(\vec{x}^*)$ définit un "voisinage" de $\vec{x}^*$.[8]

1.4 Dominance et optimalité de Pareto

Dans cette section nous allons introduire les notions liées à l'optimisation multi-objectif sur lesquelles toute la suite de ce travail va se baser.

1.4.1 Définition de la dominance

La solution $x^{(i)}$ du problème (1.1) est dite dominer une autre solution $x^{(j)}$, si les conditions suivantes sont vérifiées :

- $f_m(x^{(i)}) \leq f_m(x^{(j)}) \forall m \in \{1, ..., M\}$.
- $\exists m \in \{1, ..., M\}$ tel que $(f_m(x^{(i)}) < f_m(x^{(j)})$.

Si la solution $x^{(i)}$ domine la solution $x^{(j)}$, nous allons écrire $x^{(i)} < x^{(j)}$.

1.4.2 Propriétés de la relation de dominance

La relation binaire de dominance $\prec$, telle qu'elle est définie ci-dessus[1] :
– n'est pas réflexive, car une solution ne se domine pas elle-même ;
– n'est pas symétrique, car on n'a jamais $(x^{(i)} \prec x^{(j)})$ et $(x^{(j)} \prec x^{(i)})$;
– n'est pas antisymétrique, du fait de l'existence de solution Pareto équivalente ;
– est transitive, car $(x^{(i)} \prec x^{(k)})$ *et* $(x^{(k)} \prec x^{(j)})$ implique $(x^{(i)} \prec x^{(j)})$.

1.4.3 Optimalité de Pareto

1.4.3.1 • Pareto optimal

Une solution $x^* \in \Omega$ est Pareto optimale si et seulement si il n'existe pas une solution $x \in \Omega$ tel que x domine x^*. [10]

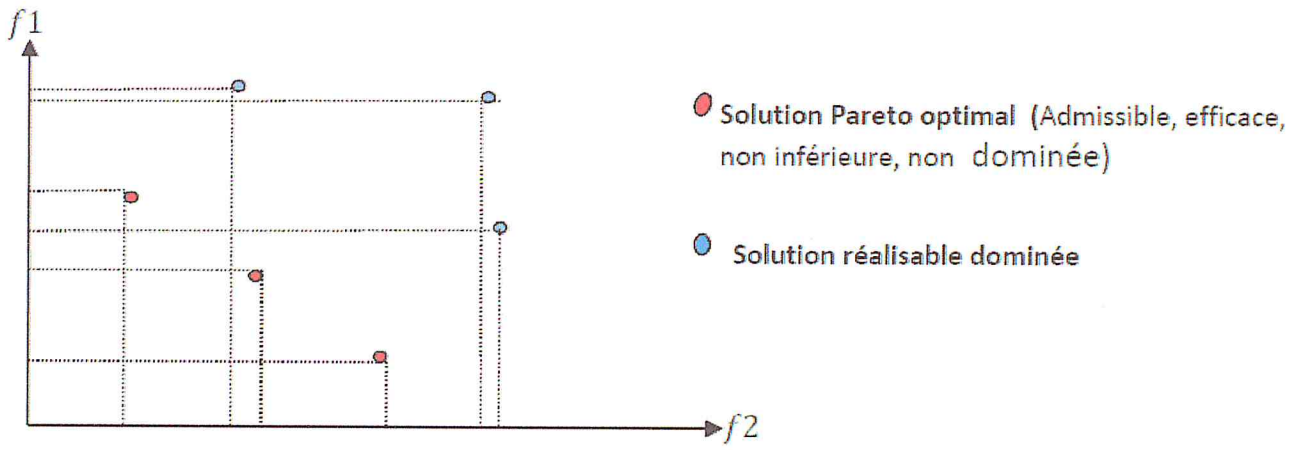


Figure 1.2 : Ensemble exact des solutions Pareto optimales

1.4.3.2 • Pareto optimalité faible

Un point $x^* \in \Omega$ est dit faiblement Pareto optimal si et seulement si $\exists x \in \Omega$, tel que $f_i(x) \leq f_i(x^*)$ pour $i = 1, ..., K$.

1.4.3.3 • Pareto optimalité forte

Un point $x^* \in \Omega$ est dit fortement Pareto optimal s'il n'existe pas un autre point $x \in \Omega$, $\vec{x}^* \neq \vec{x}$,
tel que $f_i(x) \leq f_i(x^*)$, pour $i = 1, ..., k$.

1.4.3.4 • Ensemble optimal de Pareto

Pour un problème d'optimisation multi-objectif **(MOP)** donné, l'ensemble optimal de Pareto P^* est défini comme suit :

$$P^* := \{x \in \Omega / \nexists x' \in \Omega : F(x') \leq F(x)\}.$$

1.5 Le front de Pareto

Pour un **(MOP)** donné, dont l'ensemble optimal de Pareto est P^*, le front de Pareto [10], noté PF^*, est défini comme étant :

$$PF^* := \{y \in F(x) / x \in P^*\} .$$

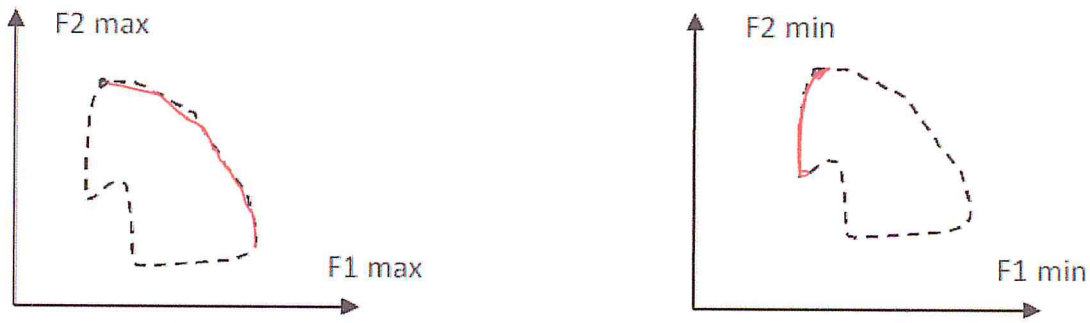


Figure 1.3 : Exemples de front de Pareto

1.6 Procédures de recherche de l'ensemble non-domine

La recherche de l'ensemble non-dominé P^* dans un ensemble fini donné de solutions P est similaire à celle de la recherche du minimum dans un ensemble fini de nombre réels. La différence entre ces deux recherches réside dans les relations

d'ordre utilisées. Dans le cas des nombre réels R l'ordre est total $\prec$, contrairement au cas des vecteurs de R^m. Dans ce dernier les éléments sont comparés au sens de la dominance de Pareto, c'est-à-dire , un ordre qui n'est pas total. Dans cette section, nous allons décrire brièvement deux procédures utilisées pour identifier P^* . Nous supposons dans la suite qu'il y a M objectifs et N points dans l'ensemble P.

1.6.1 Approche 1 : naïve et lente

Dans cette approche [1], chaque solution i est comparée á toutes les autres solutions jusqu'à ce qu'elle soit dominée par l'une d'elles. Si aucune ne la domine , elle est déclarée non dominée.

1.6.2 Approche 2 : mise à jour continue

Cette approche [1] est très similaire à la précédente, on lui a simplement rajouté une mémoire qui la rend en général plus efficace - mais pas dans tous les cas.
Chaque solution de P est comparée avec un sous-ensemble partiellement rempli et continûment mise à jour. Au début, l'ensemble P^* contient la première solution. Ensuite, chaque solution i sera comparée aux éléments de P^*. Toutes les solutions de P^* dominées par i, sont éliminées. Si aucune solution de P^* ne domine i, i est ajoutée à P^*.

1.7 Point idéal et point Nadir

On observe deux points caractéristiques associés à une surface de compromis (front de Pareto) :

- **Point idéal :**

Les coordonnées de ce point sont obtenues en optimisant chaque fonction séparément[7]. On dit aussi que les coordonnées du point idéal correspondent aux meilleures valeurs de chaque objectif(voir figure 1.4).

- **Point Nadir :**

Les coordonnées de ce point correspondent aux pires valeurs obtenues par chaque fonction objectife lorsque l'on restreint l'espace des solutions à la surface de compromis (voir figure 1.4). Le point idéal est souvent utilisé dans les méthodes d'optimisation comme point de référence. Le point nadir lui sert á restreindre l'espace de recherche, il est utilisé dans certaines méthodes d'optimisation interactive.

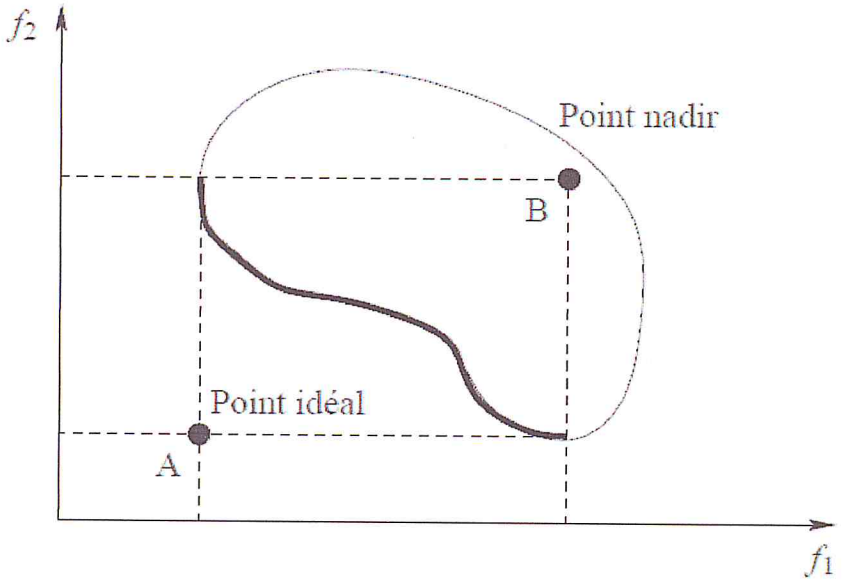


Figure 1.4 : Représentation de point idéal et point nadir

1.8 Les méthodes d'optimisation

Les méthodes d'optimisation peuvent être divisées en deux catégories :
les méthodes déterministes et les méthodes stochastiques (probabilistes) [20].
Les algorithmes déterministes (Exacts) sont le plus souvent utilisés si une relation claire entre les caractéristiques des solutions possibles et leur utilité pour un problème donné existe. Car dans ce cas, l'espace de recherche peut efficacement être exploré en utilisant par exemple un schéma du genre diviser pour mieux régner (Branch and Bound). Dans le cas contraire ou si la dimension de l'espace de recherche est très élevée, il devient plus difficile à résoudre avec une procédure déterministe.
Ainsi, les algorithmes stochastique (Approché) entrent en jeu. Un famille particulièrement pertinente des algorithmes probabilistes sont des approches fondées sur les méthodes de Monté Carlo. Ils font compenser l'exactitude de la solution par la vitesse de la recherche. Cela ne signifie pas que les résultats obtenus, en utilisant ces procédés stochastiques sont incorrects, car d'autre part, une solutions moins bonne par rapport à la meilleure possible, est plus efficace que celle qui nécessite 10^{100} années à trouver.

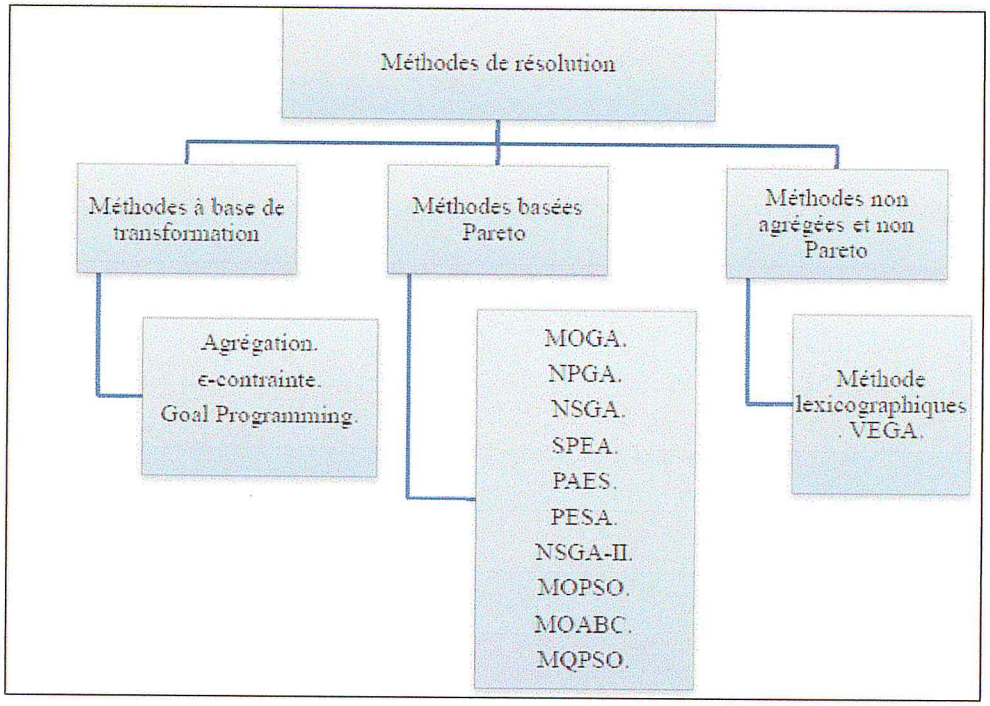


Figure 1.5 : Classification des méthodes d'optimisation

1.9 Approches de résolution des problèmes d'optimisation multiobjectifs

Parmi ces méthodes en trouve :

1.9.1 Approche basée sur la transformation du problème en un problème mono-objectif

Dans la résolution de **(MOP)**, plusieurs méthodes traditionnelles transforment le **(MOP)** en un probléme mono-objectif.

1.9.1.a Les méthodes de pondération

C'est l'une des premières utilisées pour la génération de solutions Pareto optimales. Elle consiste à transformer le problème **(MOP)** en un problème (MOP_λ) en combinant les différentes fonctions coûts f_m dans une seule fonction objectife f, généralement de façon linéaire :

$$F(x) = \sum_{m=1}^{M} \lambda_m f_m(x)$$

où les $\lambda_m \in [0,1]$ et $\sum_{m=1}^{M} \lambda_m = 1$. différents poids peuvent fournir des solutions différentes, car il arrive qu'une même solution soit générée par des poids qui ne sont pas identiques. Les résultats obtenus dans la résolution du problème (MOP_λ) dépendent

fortement du vecteur paramètre λ. Le choix des composantes de λ est fonction des préférences associées aux différents objectifs, ce qui est une tâche délicate.

Sur la figure 1.6, S est l'ensemble des valeurs du couple (f_1, f_2) respectant les contraintes définies par les droites $L1$ et $L2$ associées respectivement à deux couples de coefficients de pondération (λ_1, λ_2) différents. Cette méthode consiste "à faire tangenté" la droite $L1$ et la droite $L2$ avec l'ensemble S. Le point tangente est alors la solution recherchée. Si l'on répète ce processus pour plusieurs valeurs des coefficients de pondération, les différentes solutions trouvées forment la surface de compromis.

L'avantage de ces approches est la production d'une seule solution et ne nécessitent pas d'interaction avec le décideur. Cependant la solution trouvée peut ne pas être acceptable. L'espace de recherche est réduit de façon prématurée avant que des informations suffisantes soient disponibles. L'autre problème avec cette approche est la détermination des poids, sans avoir des connaissances sur le problème traité.

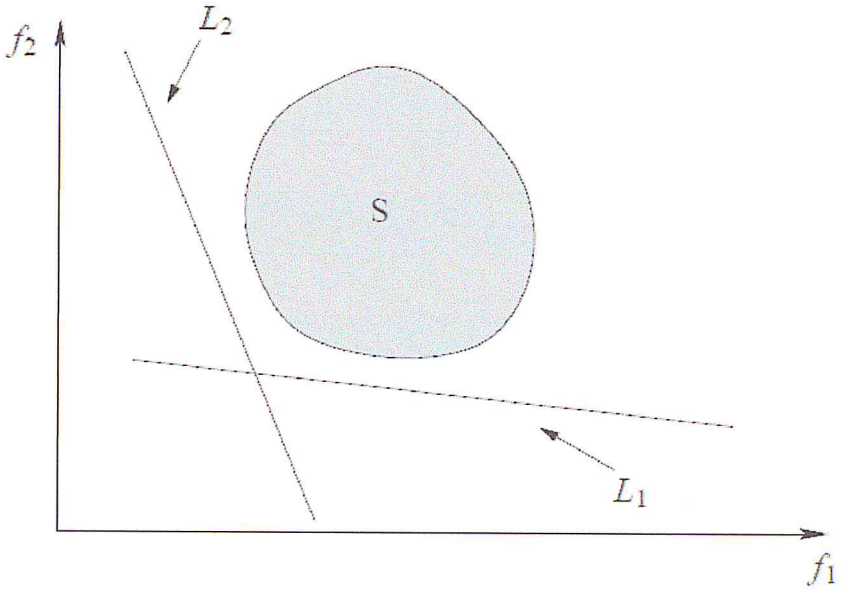


Figure 1.6 : La mèthode de pondération des fonctions objectives

1.9.1.b La mèthode ε-contraintes

Dans cette approche, le problème consiste à optimiser une fonction $f_k(x)$ sujette à des contraintes sur les autres fonctions .

$$MOP(\varepsilon) = \begin{cases} Minimiser f_m(\overrightarrow{x}) \\ x \in C \\ s.c\ f_j(x) \leq \varepsilon_j, j = 1, 2, ...M \quad \text{où } j \neq k \end{cases}$$

où $\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_{k-1}, \varepsilon_{k+1}, ..., \varepsilon_M)$.
Ainsi, un problème mono-objectif (objectif $f_k(x)$) sujet à des contraintes sur les autres objectifs est résolu. Déférentes valeurs ε_j peuvent être données pour pouvoir générer différentes solutions de Pareto optimales. La connaissance á priori des

intervalles appropriés pour les valeurs ε_j est requise pour tous les objectifs. Pour pouvoir définir les valeurs adéquates pour ε_j . Le vecteur idéal doit être calculé pour déterminer les bornes inférieures. On aura donc :

$$\varepsilon_j \leq f_j(x^*)\ j = 1, 2, ..., k-1, k+1, ..., M.$$

La méthode ε-contrainte génère généralement des solutions faiblement Pareto optimales. Cependant, si la solution optimale est unique, alors la solution trouvée devient fortement Pareto optimale. La génération de plusieurs solutions nécessite de multiples exécutions de l'algorithme avec différentes contraintes, ce qui est un exercice coûteux en termes de calcul.

1.9.1.c Programmation par but

Dans cette méthode, le décideur doit définir les but ou les références qu'il désire atteindre pour chaque objectif. Ces valeurs sont introduites dans la formulation du problème, la transformation en un problème mono-objectif. Par exemple, la fonction coût peut intégrer une norme pondérée qui minimise les déviations par rapport aux buts. Le problème peut être formulé de la manière suivante :

$$(MOP(\varepsilon; z)) \begin{cases} Minimiser(\sum_{m=1}^{M} \lambda_m |f_m(x) - z_m|^p)^{1/p} \\ x \in C \end{cases}$$

Où $1 \leq p \leq +\infty$. Les approches basées sur la programmation par but trouvent une solution non dominée si le but est choisi dans un domaine réalisable.

Cependant, le décideur est chargé de choisir le vecteur but et le poids associée à chaque objectif, ce qui est une tâche difficile sans connaitre la structure de l'espace de recherche . Si le domaine réalisable n'est pas facile à approcher, la méthode peut être inefficace.

1.10 Conclusion

On a essayé dans ce chapitre de présenter un état de l'art sur l'optimisation multi-objectife. Parmi les méthodes de résolution, on trouve l'approche qui consiste à transformer le problème multi objectif du départ **(MOP)** en un problème mono-objectif. Cela est un processus à répétition souvent très long qui demande des relations claires entre les caractéristiques des solutions possibles et nécessite une connaissance à priori sur les préférences du décideur. Tout cela consolide la tendance qui stipule de résoudre le **(MOP)** dans sa forme initiale. Ainsi, cela incite à l'utilisation de l'approche stochastique et plus particulièrement les algorithmes évolutionnaires qu'on présentera en détails dans le chapitre suivant.

CHAPITRE 2

EVOLUTION ARTIFFICIELLE

2.1 Introduction

Les algorithmes évolutionnaires font partie de la classe des algorithmes probabilistes inspirés du principe de l'évolution naturelle. Ce type d'algorithmes suit une stratégie de recherche stochastique sur une population d'individus où chaque individu représente une solution possible du problème à résoudre. Les algorithmes évolutionnaires commencent avec un ensemble de solutions aléatoires.

Pour chacune de ces solutions une valeur de fitness est évaluée et qui peut être la valeur de la fonction à optimiser.

2.2 Vocabulaire et principe de fonctionnement

Nous présentons dans cette section le vocabulaire spécifique aux algorithmes évolutionnaires, inspiré du paralléle réalisé avec les principes de l'évolution naturelle.
• Les points de l'espace de recherche Ω sont appelés " individus".
• un ensemble fini d'individus est appelé "population".
• La fonction objectife à optimiser est appelée fonction performance, ou fonction fitness.
• Le calcul de la performance d'un individu est appelé évaluation ;
• La génération correspond à une population en une certaine itération.
• Les évolutions un processus itératif de recherche des individus optimaux.
• Les opérateurs de variation sont utilisés pour générer de nouveaux individus et sont le plus souvent catégorisés en deux types d'opérateurs :
Le croisement qui consiste à échanger des parties composantes (génes) entre deux ou plusieurs individus, et la mutation qui consiste à la modification d'un ou plusieurs génes d'un individu.
• La sélection est le processus de choix des individu basée sur leur performance.
• Le remplacement est le processus de formation d'une nouvelle population à partir des parents et des enfants.

2.3 Principes de fonctionnement des algorithmes évolutuionnaires

Selon le schéma présenté par la Figure 2.1, le principe de fonctionnement d'un **(AE)** est extrêment simple[19]. On part d'un ensemble initial d'individus(chacun correspondant à une valeur du jeu de paramètre), nommé "population initiale de parents " et générée le plus souvent d'une manière aléatoire dans l'espace de recherche. Ensuite, on évalue de manière exacte la performance de chaque individu en calculant la valeur de la fonction coût correspondant à ce jeu de paramètres. L'application des trois opérateurs génétiques ("sélection, croisement et mutation ") permet de créer un nouvel ensemble d'individus appelé "population des enfants. Cette population est évaluée à son tour pour indiquer la performance de chacun de ses individus. Cette connaissance permet de décider lesquels des individus enfants méritent de remplacer certains parents. La nouvelle population constitue la "population parent" de la nouvelle génération. Si le critère d'arrêt (objet de notre projet pour le multi-objectif) est vérifié, on considérer les solutions obtenues comme

satisfaisantes, dans le cas contraire , on recommence le cycle jusqu'à satisfaction du critère en question. Le critère d'arrêt dans les **(AE)** peut être la convergence de l'ensemble des individus de la génération courante vers um même extremum, dans le cas d'un problème mono-objectif, ou alors vers un même ensemble optimal de Pareto, (difficile á obtenir dans le cas d'un MOP continu), dans le cas multi-objectif. Le plus souvent, l'algorithme est arrêté au bout d'un certain nombre d'itérations (génération) fixé a priori.

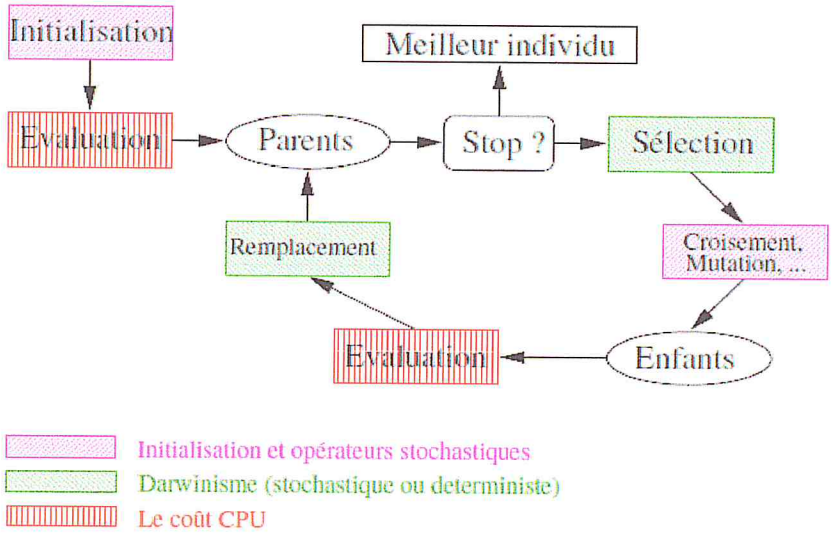


Figure 1.7 : Cycle d'un Algorithme Evolutionnaire (AE)

2.4 Les catégories principales d'algorithmes évolutionnaires

On distingue quatre grandes familles historiques d'algorithmes et les différences entre elles ont laissé des traces dans le paysage évolutionnaire actuel, en dépit d'une unification de nombreux concept.

2.4.1 Les Algorithmes Génétiques (Genetic Algorithms (GA))

Ce sont probablement les algorithmes les plus connus et utilisés dans le calcul évolutionnaire[17]. Ils ont été développés dans les années 60 pour étudier le processus complexe d'adaptation des espèces naturelles.

2.4.2 Les Stratégies d'évolution (Évolution Strategies (ES))

Ont été développés pour résoudre des probèmes numériques d'optimisation dans l'espace á paramètres réels.[18]

2.4.3 La Programmation évolutionnaire (Evolutionary Programming (EP))

Elle est utilisée dans l'espace des automates à états finis pour la prédiction de séries temporelles[15].

2.4.4 La Programmation Génétique (Genetic Programming (GP))

Apparue initialement comme sous domaine des AGs[16], PG est devenu une branche à part entière (conférence,journal,..). La spécificité de cette technique consiste à faire évoluer des structures en arborescences représentant des programmes complets .

2.5 Représentation

- **le codage binaire**

Lorsque la solution peut être représentée par une chaine de bits (nombres binaires), ou chaque gène peut prendre seulement les valeurs 0 ou 1.

- **le codage réelle**

Comme dans notre cas, nous, nous intéressant à l'optimisation multiobjective de fonctions continues[3], alors chaque individu dans l'espace de recherche est représenté par le vecteur de ses coordonnées dans R^N
Cette représentation a été introduite initialement pour les stratégies d'évolution, mais son utilisation s'est étendue rapidement aux autres types d'algorithmes évolutionnaires. Donc un individu I dans R^n ou une partie S de R^n est représenté par :

$$I : \vec{x} = (x_1, ..., x_n) \in R^n.$$

2.6 Croisement

2.6.1 le Croisement réel

Dans la représentation réelle, il y a deux manières de combiner deux allèles de deux parents :
choisir l'un des deux alléles (croisement discret) en combinat linéairement les deux (croisement intermédiaire ou arithmétique). Dans le deuxiéme cas, plusieurs variantes ont été proposées.

2.6.1.1 • Croisement discret :

$\forall i \in [1, ..., n], x_i^{'} = x_{S,i}$ ou $x_{T,i}$ oú n est la taille du vecteur réel avec S et T deux individus sélectionnés á partir de la population des parents pour l'ensemble des composantes de $\vec{x}$.

2.6.1.2 • Croisement intermédiaire :

Ce type de croisement à découpage de chromosomes est très efficace pour les problémes discrets. Ce croisement utilise deux individus S et T sélectionnés dans la population en question et soit α une variable aléatoire uniforme appartenant à l'intervalle $[0, 1]$. Ce croisement donne naissance á un nouveau élément dont les composantes sont données par :

$$\forall i \in [1, n], x_i^{'} = \alpha x_{S,i} + (1 - \alpha) x_{T,i}.$$

2.6.2 Croisement binaire simulé (SBX) :

A partir de deux parents $X = (x_1, x_1, ..., x_n)$ et $Y = (y_1, y_1, ..., y_n)$, le croisement SBX génère deux solutions nouvelles $Z^1 = (z_1^1, z_2^1, ..., z_n^1)$ et $Z^2 = (z_1^2, z_2^2, ..., z_n^2)$ de la manière suivante [20] :
Étape 1 : Générer aléatoirement une variable aléatoire uniforme u dans $[0, 1]$
Étape 2 : Calculer le nombre β selon l'expréssion :

$$\beta = \begin{cases} (2u)^{\frac{1}{1+\eta_c}}, & si\ 0.5 > u > 0 \\ (\frac{1}{2(1-u)})^{\frac{1}{1+\eta_c}}, & sinon; \end{cases}$$

Étape 3 : Déterminer les nouvelles solutions en utilisant :

$$z_i^1 = 0.5[(1 + \beta)x_i + (1 - \beta)y_i]$$

$$z_i^2 = 0.5[(1 - \beta)x_i + (1 + \beta)y_i]$$

où η_c désigne l'indice de distribution de l'opérateur SBX.

2.7 Mutation

La mutation introduit des changements aléatoires dans la structure de la population. Elle crée ce qu'on peut appeler des erreurs de recopie. Les individus doivent être reportés dans la population suivante , mais avec de légérs changements au niveau de leurs éléments. Contrairement au croisement, la mutation permet ainsi, d'explorer l'espace de recherche, alors que le précédent opérateur ne fait que l'exploiter. La mutation permet á un AG d'échapper aux optima locaux et permet de crée des éléments originaux. Si par contre, elle mute un individu vers une solution moins bonne, cette dernière est éliminée. Bien qu'elle permette á l'AE de sortir des extremums locaux, elle est moins fréquemment appliquée. C'est d'ailleurs la raison pour laquelle la mutation a un aspect marginal. Sa probabilité est en effet très faible devant celle

du croisement. La littérature affiche plusieurs types de mutation selon la nature du codage utilisé. Pour le codage réel, on peut citer entre autre :

2.7.1 La Mutation Gaussienne

Le principe de l'opérateur de mutation réelle consiste généralement à ajouter une perturbation arbitraire Gaussienne aux différentes composantes de l'individu $X = (x_1, x_2, .., x_n)$, oú n est le nombre de composantes de X. Chaque composante devienne alors :

$$x_i = x_i + \sigma N(0.1)$$

où σ est l'écart type de la mutation et $N(0.1)$ représente la loi normale centrée reduite. La difficulté de cette approche est l'ajustement des déviations standards des variables Gaussiennes utilisées. En effet, d'une part si la déviation standard est trop petite, les déplacements dans l'espace de recherche sont insuffisants pour permettre à l'algorithme de quitter le voisinage d'un optimum local. Souvent cela ne conduit pas à l'exploration de nouvelles régions de l'espace de recherche, par contre, si l'écart est élevé, l'algorithme pourra accéder à une région contenant l'optimum, mais la qualité de convergence ne sera pas satisfaisante.

2.7.2 La Mutation Polynomiale

La mutation polynomiale est très proche de la mutation normale. Pour une solution x, la solution mutée $\acute{x}$ pour une variable particuliére x_i est crée en faisant :

$x_i^{'} = x_i + \Delta_{max}\sigma_q$
Où Δ_{max} est la variation et σ_q le coefficient de variation, calculé de la manière suivante :

$$\begin{cases} \delta_L = (2u)^{\frac{1}{(1+\eta_m)}} - 1, & si\ 0 < u \leq 0.5 \\ \delta_R = 1 - (2(1-u))^{\frac{1}{(1+\eta_m)}}, & si\ u > 0.5 \end{cases}$$

Où u est un nombre aléatoire tire uniformément dans l'intervalle $[0, 1]$, et $\eta_m,)$ l'indice de distribution.

2.8 Darwinisme artificiel

2.8.1 Procédures de sélection

La partie darwinienne de l'algorithme comprend les deux étapes de sélection et de remplacement. On distingue deux catégories de procédure de sélection ces deux étapes sont totalement indépendantes de l'espace recherche[4]. Les procédures de sélection les plus fréquemment utilisées :

2.8.1.1 Sélection proportionnelle(par roulette)

La première sélection mis en place pour les AE est le tirage à roulette. La roue est divisée en secteurs et chaque secteur est associé à un individus, la superficie du secteur étant proportionnelle à la valeur de fitness de l'individu, les individus les mieux en forme auront une probabilité plus grande pour être choisis.

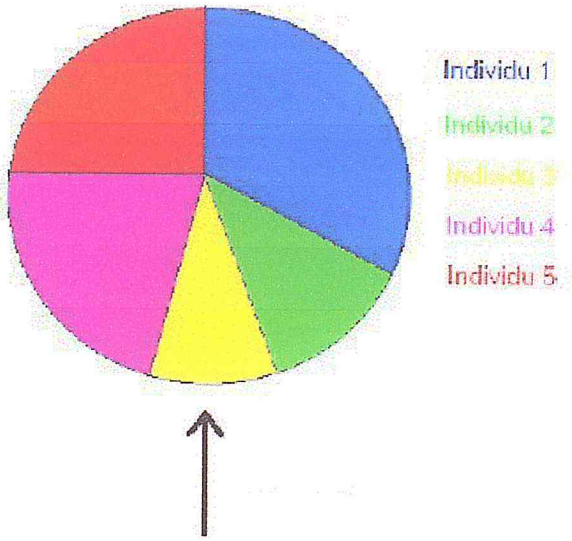


Figure 2.2 : la sélection par roulette

2.8.1.2 Sélection par tournoi

Ce principe est l'un de ceux qui donnent les meilleurs résultats. il consiste à choisir aléatoirement K individus. les performances de chacun de ces individus sont évaluées et on choisit le meilleur . On répéte ce processus jusqu'à atteindre le nombre désiré d'individus.

2.8.2 Remplacement

Cet opérateur est le plus simple, son travail consiste à réintroduire les descendants obtenus par application successive des opérateurs de sélection, de croisement et de mutation (la population $P^{'}$) dans la population de leurs parents (la population P). On trouve essentiellement deux schémas de remplacement différents :

2.8.2.1 Le remplacement déterministe :

Son caractère purement d'déterministe lui donne un rôle clef dans l'évolution vu qu'il guide la recherche vers les zones des meilleures individus. Il opére en sélectionnant les $\mu(1 < \mu \leq \lambda)$ meilleures solutions parmi :
- L'union de μ parents et λ enfants.
- L'ensemble de λ enfants.

Le premier remplacement est élitiste, et garantit une amélioration monotone de la

performance de la population, mais il s'adapte mal á un éventuel changement d'environnement. Par contre avec le second, la meilleure performance peut décroître, mais alors l'algorithme est plus flexible avec le changements d'environnement.

2.9 Algorithme évolutionnaires multi-objectif (AEMO)

Les algorithmes évolutionnaires manipulent une population de solutions au lieu d'une seule solution, comme dans le cas de l'utilisation de la plupart des autres méthodes d'optimisation.dans cette mémoire on utilisé la méthode de NSGA-II.

2.9.1 NSGA-II

Le NSGA-II est une version perfectionnée du NSGA (Nondominated Sorting Genetic Algorithm) proposé par Deb et al. Cette nouvelle version corrige toutes les critiques faites sur NSGA telles que : la complexité, non élitiste et utilisation du sharing. Le NSGA-II intégre un opérateur de sélection basé sur le crowding (calcul de la distance du surpeuplement) qu' est différent du sharing (utilisé dans le NSGA). Comparativement au NSGA , le NSGA-II obtient de meilleurs résultats sur toutes les instances présentées dans les travaux de K.Deb ,ce qui fait de cet algorithme un des plus utilisées aujourd'hui. Le NSGA-II est un algorithme élitiste, c'est á dire : qu'il n'utilise pas d'archive externe pour stoker l'élite (les meilleurs individus). Pour gérer l'élitisme , NSGA-II assure qu'à chaque nouvelle génération, les meilleurs individus rencontres soient conservés.

2.9.1.1 Une itération de NSGA-II

Aprés initialisation aléatoire de la population initialie P_0, une itération de NSGA-II se déroule [10] comme suit :

1) Créer Q_t a partir de P_t en utilisant la sélection par tournoi et en appliquant les opérateurs de variation génétique aux individus gagnants.

2) Réunir les population des parents et des enfants $R_t = P_t \cup Q_t$. Trier selon la relation de dominance de Pareto l'ensemble résultant R_t en sous-ensembles F_i.

3) Soit une nouvelle population $P_{t+1} = \emptyset$. Soit le compteur des Sous-ensembles non-dominés $i = 1$.

4) Tant que $|P_{t+1}| + |F_i| < N$ faire : $P_{t+1} = P_{t+1} \bigcup F_i$ et $i = (i + 1)$.

5) Ordonner l'ensemble F_i selon les "distances de surpeuplement" et inclure $N - P_{t+1}$ solutions ayant les valeurs de distance les plus grandes dans la population P_{t+1}.

Il est important de noter que le tri de R_t en sous-ensembles non-dominés fait en (1) et le remplissage de la population P_{t+1} peuvent être effectuer simultanément. Chaque fois qu'un nouveau front est trouvé, on vérifie s'il peut renter entièrement dans P_{t+1} . Si ce n'est pas le cas, le processus du ranking entre en jeu.

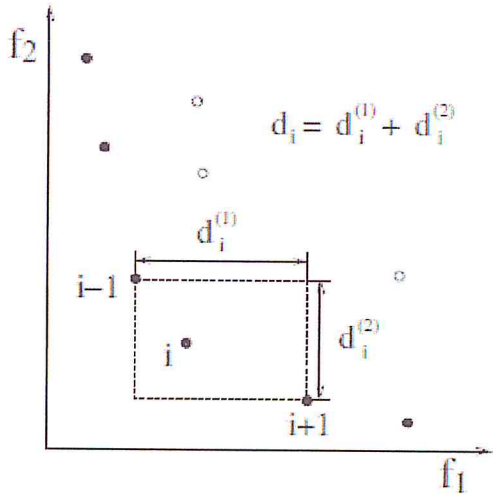


Figure 2.3 : crowding distence

Calcul des distances de surpeuplement (crowding)

Pour estimer la densité des solutions voisines [1] d'une solution i dans un ensemble non-dominé F, la quantité d_i, représente la distance de surpeuplement, dont l'algorithme est donné ci dessous :

1) soit $l = |F|$ soit d'abord $d_i = 0$ pour tout solution de F soit le compteur d'objectif $m = 1$.

2) pour l'objectif m, réordonner l'ensemble F de facon que les valeurs de f_m sur ses éléments diminues soit $I^m = sort_{[f_m>]}(\mathbf{F})$ le vecteur des indices. C'est -à-dire I^m dénote l'indice de la solution i dans la liste ordonnée selon l'objectif m

3) Pour chaque solution i telle que $2 \leq I^m \leq (l-1)$, mettre à jour la valeur de d_i comme suite :

$$d_i \longleftarrow d_i + \frac{f_m^{I_i^m+1} - f_m^{I_i^m-1}}{f_m^{max} - f_m^{min}}.$$

et associer les valeurs de distance trés grandes aux solutions sur les extrémités de F c'est-à-dire :
$I_i^m = 1$ ou $I_i^m = l$, $d_i = inf$.

4) Si $m = M$, la procédure est terminée, sinon incrémenter le compteur d'objectifs $m \longleftarrow (m+1)$ et retourner a l'étape 2.

La quantité d_i correspond au semi-prérimétre du cuboide dont les vertex sont les voisins les plus proches de i (voire figure 2.3).

2.10 Évaluation de la performance d'un AE

L'évaluation de la performance est spécifique dans le cas des algorithmes évolutionnaires multi-objectifs (MOEA), [8] car elle comprend deux aspects : la convergence vers la surface de Pareto et la distribution (uniforme) des solutions le long de cette surface. Ces deux aspects doivent donc étre pris en compte par les métriques distignées á mesurer la qualité des résultats obtenus par un AEMO. Dans la littérature, on trouve une panoplie de mesures répartiées en métriques de convergence, métriques de diversité, et métriques implicitement tiennent compte des deux aspects. ici, nous avons choisi de présenter les plus utilisées et qui tiennent compte des deux aspects et qui vont nous sérvir dans notre application.

2.10.1 Distance générattionnelle inversée (IGD)

Soient PF_{Pareto} un ensemble des solutions distribuées de manière uniforme sur le vrai front de Pareto et PF^* ensemble de solutions trouvé par le MOEA. on designe par v les élements du PF_{Pareto} et par X ensemble des solutions non dominée dans PF^* .

$$IGD = \frac{\sum_{v \in PF_{Pareto}} d(v, X)}{|PF_{Pareto}|}$$

$d(v, X)$, désigne la distance euclidienne minimum entre v et les point de X. Pour avoir une faible valeur d'IGD. L'ensemble X devrait être proche de PF_{Pareto}.

2.10.2 Indicateur d'hypervolume

Cette métrique (Hyperarea and Ratio) permet de mesurer la surface occupée par l'ensemble de solutions X dans PF^*. Sa définition est la suivante :

$$HV = \left\{ \bigcup_i a_i | x_i \in X \right\}.$$

avec :
a_i : surface occupée par la solution x_i (se reporter à la figure 2.3 pour une illustration de cette surface). Le calculer de cette métrique dans le cas d'un ensemble de solutions X en deux dimensions est donné par la procédure ci-dessous :
Etape 1 : Extraire la surface de compromis de l'ensemble de solutions PF^*. On appelle X cette surface de compromis.
Etape 2 : Ranger les éléments de X selon l'ordre croissant par rapport au premier objectif f_1.
Etape 3 : Appliquer l'algorithme suivant :
1. $i = 1$, $HV = f_1(x_i) * f_2(x_i)$
2. $i = i + 1$
3. si $i \leqq N$, $HV = HV + [f_1(x_i) - f_1(x_{i-1})] * \Delta f_2(x_i)$ allez en 2.
4. si non $(i > N)$ alors stop
avec :
N : nombre d'éléments dans X,
x_i : un élément dans X
On peut aussi définir tau d'hyper-volume HR :

$$HR = \frac{HV}{HVT}.$$

avec :
HV : surface occupée par l'ensemble de solutions X ;
HVT : surface occupée par les solutions les plus proches á X dans le vrai front de Pareto PF_{Pareto}.

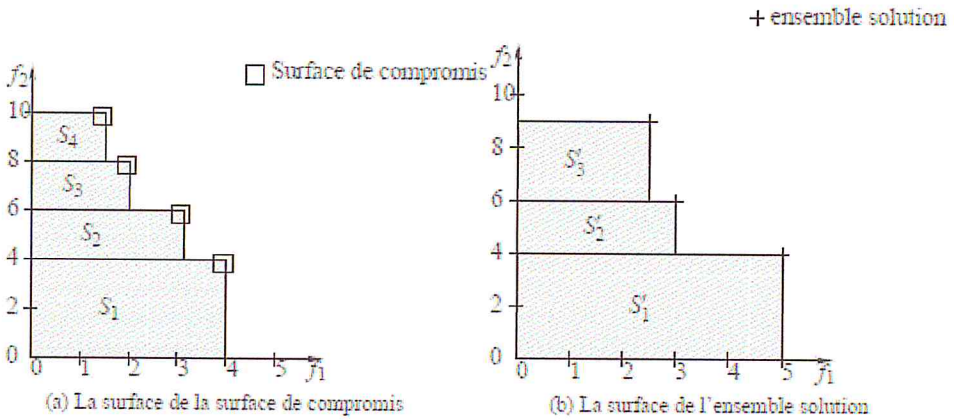


Figure 2.4 : Le calcul des surfaces

2.11 conclusion

Nous avons présenté un état de l'art sur les algorithmes évolutionnaires d'un manière générale dans ce deuxième chapitre.
Après avoir rappelé quelque notion de l'algorithme évolutionnaire nous passé en revue les principaux algorithmes MOEA de l'état de l'art ainsi que les deux types de codage pour représenter les solutions dans la population. On a parlé aussi sur la partie Darwienne de l'algorithme la sélection et le remplacement, ainsi que les indicateurs de performance et procédures de comparaison.

CHAPITRE 3

LE CRITÈRE D'ARRÊT DANS LES MOEA ET LA THÉORIE DE L'ENTROPIE

3.1 Introduction

Lors de l'utilisation des MOEA, le nombre de génerations nécessaire pour assurer une bonne approximation du front de Pareto d'un probléme donné est une question critique qui incombe les praticiens. Dans l'absence d'un critére adéquat, la pratique courante consiste á fixé a priori le nombre d'itérations ou á exécuter l'algorithme jusqu'á épuisement du temps disponible. Cependant, cette approche est á double tranchant, dans le sens, qu'un MOEA peut fournir des solutions sous-optimales si le nombre de générations a priori fixées est insuffisant ou le temps de calcul disponible est court. En revanche, un grand nombre de générations , ou un temps de calcul trop long peut entraîner un gaspillage de ressources. De plus, ces critéres sont dépourvus de toute intelligence intrinséque comme par exemple, une évaluation intermittente ou finale de la qualité de l'approximation obtenue du PF par rapport au véritable front. Compte tenu de cela, ce chapitre sera organisé comme suite : nous commencant par rappeler les caractééristiques d'un critére d'arrêt robuste, suivé d'un rappel sur le concept de l'entropie, introduit par Shannon [12] pour la théorie de l'information. Ensuite, nous allons voire comment définir un critére d'arrêt pour les MOEA basé sur l'entropie.

3.2 Théorie de l'Entropie

Le terme entropie a été introduit en 1865 par Rudolf Clausius á partir d'un mot grec signifiant "transformation". Il caractérise le degré de désorganisation ou de manque d'information d'un système. En théorie de l'information, l'entropie (de Shannon), due à Claude Shannon, est une fonction mathématique qui, intuitivement, correspond à la quantité d'information contenue ou délivrée par une source d'information.

3.2.1 Définitions et Proprietés

a) Définition

Pour une source, qui est une variable [6] aléatoire discrète X comportant n symboles, chaque symbole x_i ayant une probabilité $P(x_i)$ d'apparître, l'entropie H de la source X est définie comme :

$$H(x) = -\mathbb{E}[log(P(x))] = -\sum_{i=1}^{n} P(x_i) log P(x_i).$$

où $\mathbb{E}$ désigne l'espérance mathématique. L'entropie H mesure l'incertitude associée à La prédiction du résultat d'une variable aléatoire.

Si on dispose de deux variables aléatoires X et Y, on définit d'une facon analogue la quantité $H(X,Y)$, appelé l'entropie conjointe, des variables X et Y :

$$H(X,Y) = -\sum_{i,j} P(X = x_i, Y = y_j) \log P(X = x_i, Y = y_j).$$

Ainsi que l'entropie conditionnelle de Y relativement à X :

$$H(Y \mid X) = -\sum_{i,j} P(X = x_i, Y = y_j) \log P(Y = y_j \mid X = x_i) =$$
$$\sum_i P(X = x_i) \left(-\sum_j P(Y = y_j \mid X = x_i) \log = P(Y = y_j \mid X = x_i) \right)$$

b) Propriétés

Voici quelques Propriétés importantes de l'entropie de Shannon :

- $H(X) \geq 0$ avec égalité si et seulement s'il existe i tel que $P(X = x_i) = 1$
- $H(X) = -\sum_i p_i \log p_i \leq -\sum_i p_i \log q_i$ où q_i est une distribution de probabilité quelconque sur la variable X (Inégalité de Gibbs).
- $H(X) \leq \log(n)$. Avec égalité si et seulement si la loi de probabilité de x est uniforme.
- Elle est symétrique : $H(X, Y) = H(Y, X)$
- Elle est continue
- $H(X, Y) = H(X) + H(Y \mid X)$
- $H(X, Y) \leq H(X) + H(Y)$ avec égalité si et seulement si les variables sont indépendantes.
- $H(Y \mid X) \leq H(Y)$
- $H(Z \mid X, Y) \leq H(Z \mid X)$
- $H(X_1, \ldots, X_n) = H(X_1) + H(X_2 \mid X_1) + \ldots + H(X_n \mid X_1)$
- $H(X_1, \ldots, X_n) \leq \sum_{i=1}^{n} H(X_i)$

3.2.2 Divergence de Kullback-Leibler(Entropie Relative)

En théorie des probabilités et en théorie de l'information [12], la divergence de Kullback-Leibler (ou divergence K-L ou encore entropie relative) est une mesure de dissimilarité entre deux distributions de probabilités $P(x_i)$ et $Q(x_i)$. Ce concept peut être utilisé pour comparer deux processus stochastiques. Dans le cas discrêt, cette mesure est donnée par :

$$KL(p||q) = -\sum_{i=1}^{n} P(x_i) log(p(x_i)/q(x_i)).$$

Propriétés

1) $KL(p||q)$ est toujours non négatif, c'est-à-dire : $KL(p||q) \geq 0$.
2) Elle n'est pas symétrique puisque $KL(p||q) \neq KL(q||p)$
3) Si nous avons : $P(x) = Q(x)$ alors cela implique :$KL(p||q) = KL(q||p)$

Ce concept de divergence de Kullback-Leibler(Entropie Relative) sera utilisé pour la détection en ligne du nombre de générations suffisants afin que l'algorithme (MOEA) se stabilise. Cette stabilisation signifie qu'une bonne approximation a été obtenue ,ou qu'elle ne peut être obtenue en raison d'une stagnation. Notre application demande un savoir faire en terme d'estimation statistique.

3.2.3 Estimation des Distributions de Probabilités

L'estimation des distributions de probabilités ($P(x_i)$ et $Q(x_i)$ nécessaire dans le calcul de l'entropie relative) peut se faire en utilisant les méthodes paramétriques, semi-paramétriques ou non paramétriques [13]. Les méthodes d'estimation non paramétriques sont les plus flexibles car elles ne font pas d'hypothèses sur la forme fonctionnelle de la distribution, et l'estimation de la densité est entièrement axée sur les données. Parmi les mééthodes les plus couramment utilisées dans cette catégorie, on trouve la Méthode des histogrammes multivariés qui utilise l'histogramme non pas comme un outil de visualisation, mais comme une estimation statistique de la distribution sous-jacente de l'échantillon.

Soit donné un espace D-dimensionnel, la mèthode d'histogramme multidimensionnelle partitionne chacune des dimensions D en un nombre fixe d'intervalles (n_b), dèfini par un point d'ancrage (souvent, l'origine) et une largeurs fixe pour chaque dimension, à savoir, $h_1, ..h_D$. Ces partitions résultent un nombre n_b^D de cellules. Chaque cellule est un hyperboxe d'hypervolume donné par $\prod_{j=1}^{D} h_j$. La fonction de distribution de probabilité associée à la cellule x_i est donnée par :

$$P(x_i) = \frac{k(x_i)}{N^*}$$

N^* : représente l'effectif total de points
$K(x_i)$: représente le nombre de point dans la cellule x_i.
De toute évidence, avec l'augmentation de la dimensions (D), le nombre de cellules n_b^D augmente de façon exponentielle et, par conséquent, la fixation de n_b (dont dépend le lissage de la distribution de probabilité) est un défi majeur associé à cette méthode.

3.3 Construction du critére d'arrêt pour les MOEA

Notre critère d'arrêt pour les MOEA sera basé sur la comparaison de deux distributions P et Q respectivement des populations $\mathbb{P}$ et $\mathbb{Q}$ qui correspondes respectivement à deux génerations successives lors de l'exécution d'un MOEA. Cette mesure de dissemblance à base d'entropie relative. Les distributions P et Q seront estimées par la mèthode des histogrammes multidimensionnelles.

3.3.1 Mise en place de la méthode d'estimation

Pour une gestion efficace des calculs sur l'espace des objectifs M-dimensionnel, on suppose que tous les hyperboxes ont la même largeur et nous proposons d'utiliser la structure de données suivantes :
1) J_I : Un ensemble de cellules(hyperboxes) représentant la région d'intersection pour $\mathbb{P}$ et $\mathbb{Q}$, où chaque cellule satisfait $P(x_i) > 0$ et $Q(x_i) > 0$.
2) P_{NI} : Un ensemble de cellules représentant la region de non-intersection, chaque cellule satisfait $P(x_i) > 0$ et $Q(x_i) = 0$.
3) Q_{NI} : Un ensemble de cellules représentant la région de non-intersection, où chaque cellule satisfait $P(x_i) = 0$ et $Q(x_i) > 0$.
4) C : Vecteur qui stocke les cellules des régions J_I et P_{NI} .
5) C_q : Vecteur qui stocke les cellules de la région Q_{NI}.

6) P_c et Q_c : Vecteurs qui stockent le nombre de solutions dans chaque cellule de C, composée d'individus dans les deux populations $\mathbb{P}$ et $\mathbb{Q}$.
7) Q_{cq} : Un vecteur qui stocke le nombre de solutions dans chaque cellule de C_q, composée uniquement par des individus de $\mathbb{Q}$.

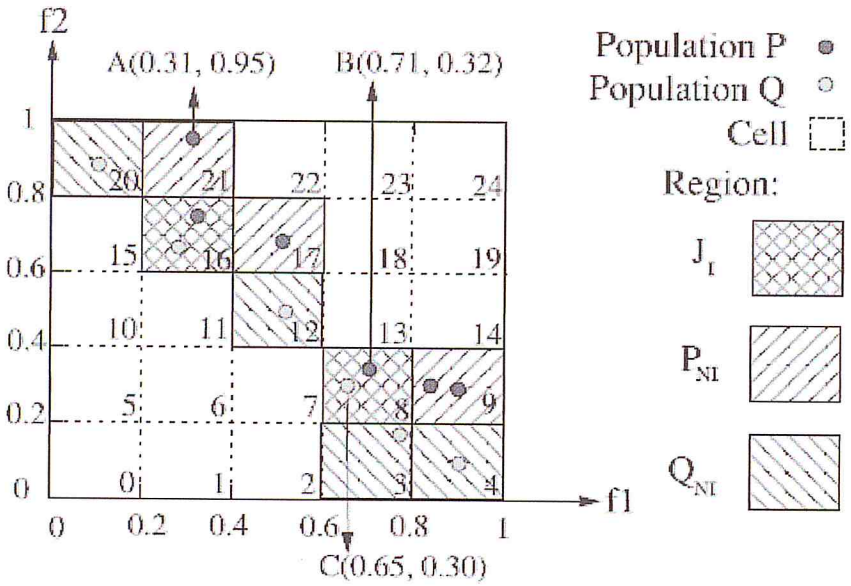


Figure 3.1 : Soulignant l'effet de la dispersion de la population sur le nombre de cellules requisespour

Exemple :
Pour un exemple d'illustration, considérons que $\mathbb{P}$ et $\mathbb{Q}$ comprennent six solutions chacune, comme le montre la Fig.3.1, notamment :
1)$|J_I| = 2$; $|p_{NI}| = 3$; $|Q_{NI}| = 4$; $|C| = 5$; $|C_q| = 4$;
2)$C = \{8, 9, 16, 17, 21\}$; $C_q = \{3, 4, 12, 20\}$;
3)$P_c = \{1, 2, 1, 1, 1\}$; $Q_c = \{1, 1, 1, 1\}$; $c = \{1, 0, 1, 0, 0\}$; $Q_{cq} = \{1, 1, 1, 1\}$.

3.3.2 Identification des Cellules (Hyperboxes)

Pour faciliter l'identification des cellules, (de C ou de C_q), lors du processus d'affectation des individus, nous assignons à chaque hyperboxe un numéro d'identification unique. Soit $S = (s_1, ...s_M)$ une solution dans $\mathbb{P}$ ou $\mathbb{Q}$, nous obtenons cette identification par la procédure suivante :
1) Soient O_{maxj} et O_{minj} respectivement la valeur maximale et la valeuri minimale par rapport au j ème objectif de $\mathbb{P} \cup \mathbb{Q}$.
2) Soit $GetCell_{id}$ la fonction qui affecte une valeur d'identification (c) à la cellule contenant la solution S, comme suit :

- Ramener les valeurs composantes de S dans [0,1] en posant :

$$\bar{s}_j = \frac{s_j - O_{min,j}}{O_{max,j} - O_{min,j}} \text{ pour } j = 1, ..., M.$$

- Soit le vecteur $B = \{(0/n_b), (1/n_b), ..., (n_b/n_b)\}$ de dimension $n_b + 1$ et qui définit un ensemble d'intervalles tels que :

$$B_{K_j} \leq \bar{s}_j \leq B_{K_{j+1}}, K_j \in [0, ..., n_b - 1]$$

- Déterminer le numéro d'identification de la cellule, par :

$$c = \sum_{j=1}^{H} k_j * {n_b}^{j-1}$$

À titre d'illustration, considérons les solutions A, B et C dans la Fig 3.1 ; où $M = 2$ et $n_b = 5$. Le numéro d'identification de la cellule de $A(k = 1, 4)$ est donné par $c = 21$, alors que celle de B et $C, (K = 3, 1)$ est $c = 8$, ce qui implique que ces deux solutions appartiennent à la même cellule (même numéro d'identification).

3.3.3 Algorithme de construction d'histogramme multidimensionnel

Les étapes globales de l' algorithme de construction d'histogramme multidimensionnel (Algorithme 1(Annexe)) sont les suivantes :
1) Trouver une cellule correspondant à chaque solution dans $\mathbb{P}$, en utilisant la fonction $GetCell_{id}$ (décrite dans la section précédente). Si la cellule figure déjà dans le vecteur C, augmenter d'une unité la composante correspondant à cette cellule dans le vecteur P_c . Sinon, procédez comme suit :

- Ajoutez la cellule dans le vecteur C, afin d'avoir une trace de toute les cellules trouvées pour la population $\mathbb{P}$.
- Mettre à jour le vecteur P_c pour la solution trouvée, en initialisant á 1 la composante correspondant á la cellule.
- initialiser a zéro la composante correspondante à cette cellule dans le vecteur Q_c (Si aucune solution de la population $\mathbb{Q}$ n'existe).

Dans l'Algorthim 1 (Annexe) cette partie correspond aux étapes 2-12.

2) Trouver une cellule correspondant à chaque solution dans $\mathbb{Q}$, en utilisant la fonction $GetCell_{id}$. Si la cellule existe déjá dans le vecteur C, augmentez la position correspondante dans le vecteur Q_c. Sinon, procédez comme suit :

- Si la cellule existe dans le vecteur Cq, incrémenter d'une unité la composante correspondante dans le vecteur Q_{cq}.
- Si non :
  - ajouter la cellule dans le vecteur Cq pour garder une trace de toutes les cellules trouvées pour la population $\mathbb{Q}$ qui n'ont pas été trouvées pour la population $\mathbb{P}$
  - Initialiser a 1 la position correspondante dans le vecteur Q_{cq}.

Dans l'Algorthime 1 (Annexe) cette partie correspond aux étapes 13-27.
En résumé, l'algorithme1 (Annexe) assure que pour $\mathbb{P}$ et $\mathbb{P}$, le nombre de solutions existantes dans chaque cellule est stocké dans les vecteurs Pc, Qc et Qcq. Cela permet de calculer la distribution de probabilité associée à chaque cellule (4), une étape qui est directement implémentée dans l'Algorithme 2 (Annexe).

3.4 Mesure de Dissimilarité

Dans cette partie, nous présentons la mesure de dissmilarité qui nous permet d'identifier en ligne le nombre de génération a partir du quel notre MOEA se stabilise.
Soit une cellule $x_i \in J_I$. Contenant des solutions de $\mathbb{P}$ et$\mathbb{Q}$, la mesure de dissmilarité $D(P,Q)_I$ est défini par :

$$D(P,Q)_I = KL(P||Q) + KL(Q||P)$$

avec :

$$KL(P||Q) = -\sum_{i=x_i \in J_I} \frac{P(x_i)}{2} \log\{\frac{Q(x_i)}{P(x_i)}\}.$$

$$KL(Q||P) = -\sum_{x_i \in J_I} \frac{Q(x_i)}{2} \log\{\frac{P(x_i)}{Q(x_i)}\}.$$

si $D(P,Q)_{Y_p}$ et $D(P,Q)_{Y_Q}$ représentent la mesure de dismilarité pour $x_i \in P_{NI}$ et $x_i \in Q_{NI}$ respectivement, la mesure de dissmilarité de la combinaison des deux ensembles disjoints est donnée par :

$$D(P,Q)_Y = D(P||Q)_{Y_P} + D(P||Q)_{Y_Q}$$

avec

$$D(P||Q)_{Y_P} = -\sum_{x_i \in P_{NI}} \frac{P(x_i)}{2} \log P(x_i)$$

.

$$D(P||Q)_{Y_Q} = -\sum_{x_i \in P_{NI}} \frac{Q(x_i)}{2} \log Q(x_i)$$

.
Finalement la mesure de dissmilarité entre deux population $\mathbb{P}$ et $\mathbb{Q}$ d'un MOEA s'écrit :

$$D(P||Q) = D(P||Q)_I + D(P||Q)_Y$$

cette mesure admet les propriétes suivantes :
1) Non négative $D(P,Q) \geq 0$.
2) Symétrique $D(P,Q) = D(Q,P)$.
3) Si $P(x) = Q(x) : \forall X\ (P(x_i) = Q(x_i))\ \forall\ x_i \in J_I$ et $P_{NI} = Q_{NI} = \varnothing|$ alors $D(P,Q) = 0$.

Ainsi, si les ensembles $\mathbb{P}$ et $\mathbb{Q}$ représentent respectivement deux générations successives de la populations du MOEA, la mesure $D(P,Q)$ peut être élevée pendant les premiéres itérations, en raison de la différence sensible des populations. Par la suite, on peut avoir :
1) des valeurs de $D(P,Q)$ élevées et variées, si les populations du MOEA continuent a avancer dans la partie non optimale de l'espace des objectifs.

2) des valeurs de $D(P,Q)$ élevées, mais presque similaires, si les populations du MOEA Stagne dans la même partie non optimale de l'espace des objectifs .
3) des valeurs de $D(P,Q)$ nulle ou proche de zéro lorsque les populations des générations successives deviennent largement similaires. Cela peut se produire dans des deux scénarios où un MOEA :
A) offre une bonne approximation PF (bonne convergence et distribution de qualité à travers la PF).
B) stagne dans une partie du vrai PF (bonne convergence et une distribution médiocre des solutions dans le PF).
Ces caractéristiques de la mesure de dissimilarité, promettent de contrer la faiblesse des testes d'arrêt habituels (la fixation du nombre de générations a priori ou l'exécuter jusqu'à épuisement du temps de calcul disponible).
La moyenne et l'écart type de la mesure de dissmilarité donné par les formules suivantes.
$M_t = D_t$ et $M_t = (1 \div t) \sum_{i=1}^{t} D_i$ si $t \geq 2$
$S_t = (1 \div t) \sum_{i=1}^{t} (D_i - M_t)^2$

3.5 Algorithme du critére d'arrêt pour MOEA

Cette partie nous présentons un algorithme du critère d'arrêt pour MOEA (Algorithme 2), intégrant : 1) l'estimation de la densité de probabilité par un algorithme d'histogrammes multidimensionnels (Algorithme 1) ; 2) calcul de la mesure de dissmilaritévet 3) un critère dárrêt basé sur la moyenne et l'écart type des mesures de dissimilarité. Alors que les éléments 1) et 2) ont déjà été discutés en détail, le critère d'arrêt est présenté ci-dessous.

3.5.1 Critère d'arrêt

Les MOEA commence par une population initiale, plusieurs générations sont obtenues par variation (Croisement et mutation) et de sélection, dans le but d'atteindre le PF. Les MOEA sont de nature stochastique et par conséquent, il est normale que la mesure de dissimilarité soit variable au cours des générations successives. En général, pour un problème donné (difficulté dépend du nombre et de la nature des objectifs, des contraintes et des variables), et selon les paramétres du MOEA , la variation est déterminante dans l'avancement de la population vers le vraie POF.

Soit i le compteur de génération, t désigne la génération actuelle, D_i représente la valeur $D(P,Q)$ de la i^{eme} génération ; et M_t et S_t désignent respectivement la moyenne et l'écart type de $D(P,Q)$ de la premiére à la t^{eme} génération, et qui sont données respectivement par :

Dans l'algorithme 2 (Annexe) le critère d'arrêt s'obtient lorsque la moyenne et l'écart type des mesures de dissimilarité pour un pré-spécifique nombre de générations successives (n_s) du MOEA soient les mêmes à un nombre prédéterminé (n_p) de décimaux prés. Dans ce cas le nombre d'itérations effectués par notre MOEA est noté N_{gt}. l'idée de base derriére ce critére d'arrêt est rationnelle car si n_p et N_{gt} sont des nombres raisonnablement importants (contextuellement, par exemple, $n_p \geq 2$ et $n_s \geq 20$), rend notre critère rigoureux dans la mesure où :
1) il ne peut pas être satisfait par un ensemble de solutions générées au hasard, et

qui sont dans la partie non optimale de l'espace des objectifs
2) il peut être satisfait dans les cas où le MOEA se stabilise, et les variations des valeurs de $D(P,Q)$ au cours des différentes générations ne soient pas assez grandes pour violer l'exigence de conformité n_p.

Les étapes de l'Algorithme 2 (Annexe) sont telles que décrites ci-dessous
1) (Étape 2) Générer une population (P) de solutions réalisables au hasards.
2) (Étape 3) soit P la population initiale, exécutez un MOEA pour générer une nouvelle population (Q) de solutions.
3) (Étape 4) En utilisant n_b et les populations P et Q comme donnée, exécutez l'Algorithme 1 et obtenir les vecteurs : C, C_q, P_c, Q_c et Q_{cq}.
4) (Étape 5) Initialiser la valeur de la mesure de dissimilarité a zéro, $D_t = 0$ (á chaque génération, t).
5)Sur la base des vecteurs calculés C, C_q, P_c , Q_c et Q_{cq} :
a) (étapes 6 à 13) Pour chaque cellule dans C : Calculer la distribution de probabilité associée à P (p, étape 7) et Q (q, étape 8), la mesure de dissmelarité en utilisant (8) ou (12) selon l'appartenance d'une cellule dans j_I ou P_{NI}, et mettre à jour D_t ;
b) (étapes 15-18) pour chaque cellule dans C_q : calculer la distribution de probabilité associée à Q (Q, étape 16), et la mesure de dissmilarité en utilisant (13), et mettre à jour D_t.
6) (Étapes 19-21) Calculer la moyenne et l'écart type [Mt et St,] de D_t, et arrondissez les deux à la dixième décimale pour déterminer M_t et S_t.
7) (étapes 22 à 31) Si le nombre de générations (t) dépasse ns, stopper le MOEA et poser : Ngt = t, si :
a) le Les valeurs entre Mt et Mt-ns sont égales ($c1 = vrai$) et
b) les valeurs entre St et St-n sont égales ($c2 = vrai$). Sinon, réglez $P = Q$ et revenez à l'étape 2.

CHAPITRE 4

EXPERIMENTATION TESTS ET RÉSULTATS

4.1 Introduction

L'objectif de ce présent chapitre est de donner une synthèse des résultats obtenus lors de nos exprémentations. Pour cela nous avons choisi d'incorporer notre test d'arrêt dans un MOEA et de le tester sur un certain nombre de problémes tests. Le choix du MOEA est porté sur le NSGA-II car cette algorithme est l'un des plus utilisés actuellement et plus particulièrement quand il s'agit de deux ou trois objectifs. Les problèmes tests considérez ici sont : 'ZDT' [13] et 'DTLZ' [14]. Les résultats obtenus vont nous permettre d'évaluer les performances de notre réalisation.

4.2 Problèmes tests

L'efficacité de l'Algorithme 2 (Annexe)est étudiée sur un large éventail de problèmes tests avec différents niveaux de difficulté, y compris, les ZDT de 1 à 4 [14] et les DTLZ 1,2, 4 et 5 [15] ; (plus de détails sur ces problèmes sont fournis dans le tableau ci-dessous).

Le nom de problème	Nombre d'objectifs (M)	Nombre de variables (n)	paramètre (K)
ZDT1	2	30	/
ZDT2	2	30	/
ZDT3	2	30	/
ZDT4	2	30	/
ZDT6	2	30	/
DTLZ1	3	$M-1+K$	K=5
DTLZ2	3	$M-1+K$	K=10
DTLZ4	3	$M-1+K$	K=10
DTLZ5	3	$M-1+K$	K=10

TABLE 4.1 – les paramètres des problèmes tests choisis.

4.3 Initialisation et paramétrage de nos Algorithmes

On s'intéressera ici essentiellement au choix

4.3.1 Paramètre de l'Algorithme 2 (test d'arrêt)

$n_b = 10$; $n_s = 10$; et $n_p = 2,\ 3$ et 4.

4.3.2 Paramètres du NSGA-II (croisement et mutation)

La probabilité de croisement est $P_c = 0.9$ et celle de mutation $P_m = 0.5$. Nous utilisons l'opérateur de croisement binaire simulé (SBX) et la mutation polynomiale dans le NSGA-II, les indices de distribution sont : $n_c = 20$ et $n_m = 20$.

4.3.3 Taille de population

La taille de la population initiale et la population actualisée pendant le processus d'évolution est $N = 100$ pour un nombre d'objectifs $M = 2$ et $M = 3$.

4.3.4 Le langage de programmation

Pour la réalisation de notre projet nous avons utilisé le langage de programmation scilab version 6.0.0, qu' est un langage interprété de type dynamique, extensible et facile à manipule. Le scilab est un logiciel libre de calcul scientifique développé depuis 1990 par l'INRIA (Institut National de Recherche en Informatique et Automatique) et L'ENPC (Ecole National de ponts et chaussées). Il est disponible gratuitement sur le site internet **www.scilab.org**. Dans sa syntaxe, son architecture et son fonctionnement, scilab est très proche du logiciel Matlab commercialisé par (Mathworks inc). Depuis 2003. Le développement de scilab se fait dans le cadre d'un groupe d'entreprises et d'organismes de recherche constitué de : Axs ingénierie, CEA, CNES, Cril Technology, Dassault-Aviation, EDF, ENPC, Esterel Technologies, INRIA, PSA Peugeot, citroën, Renault, Thales.

4.4 Résultats expérimentaux et discussion

Dans cette partie nous prsentant les performances de l'Algorithme 2 (Annexe) lors de nos expérimentations. Les valeurs de performances considérées ici sont : Le N_{gt} déterminé par l'Algorithme 2 (pour n_p = 2, 3 et 4), l'hypervolume pour les fonctions testes ZDT et la distance générationnelle inversée dans le cas des fonctions testes DTLZ .

4.4.1 Résultats expérimentaux d'hypervolume

Plusieurs essais ont été effectués en utilisant NSGA-II (MOEA), pour des valeurs de n_p et appliquée sur les problèmes ZDT. Le tableau ci dessous donne pour chaque fonction ZDT la valeur du nombre de génération N_{gt}, la mesure de dissmilarité et l'hypervolume obtenu pour chaque valeurs de n_p.

	Génération N_{gt}			**dissmilarité mesure D(p,q)**			**Hypervolume**		
n_p	$n_p = 2$	$n_p = 3$	$n_p = 4$	$n_p = 2$	$n_p = 3$	$n_p = 4$	$n_p = 2$	$n_p = 3$	$n_p = 4$
ZDT1	358	1036	3822	0.0155	0.0109	0.0102	0.9887	0.9968	0.9983
ZDT2	465	1674	3158	0.0153	0.0126	0.0048	0.9618	0.9964	0.9956
ZDT3	353	1081	4148	0.0154	0.128	0.0069	1.306	1.2582	1.3239
ZDT4	353	969	2989	0.094	0.0433	0.0115	1.008	1.0015	1.001
ZDT6	992	2402	6753	0.014	0.006	0.0051	0.6	0.9271	0.9789

TABLE 4.2 – les résultats de la mesure de dissmilarité et d' hypervolume pour n_p = 2, 3 et 4.

Pour étudier le comportement de l'algorithme en fonction de nombre d'itérations, nous avons choisi de s'intéresser à la variation de la valeur de hypervolume en fonction du nombre de générations pour chacune des fonctions ZDT considérée. Les

courbe ci-dessous montrent une allure convergente à partir d'un certain rang.

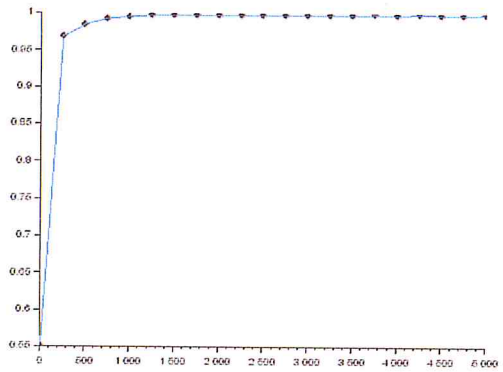


Figure 4.1. L'hypervolume en fonction du nombre d'itération dans le cas du problème ZDT1

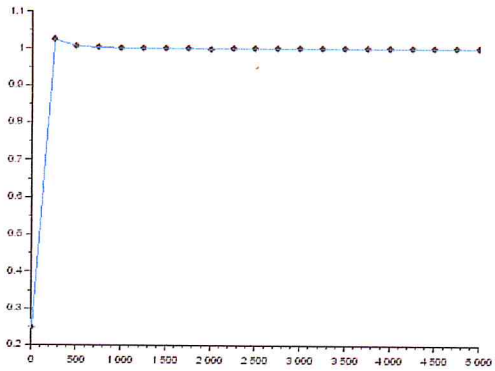


Figure 4.2. L'hypervolume en fonction du nombre d'itération dans le cas du problème ZDT2

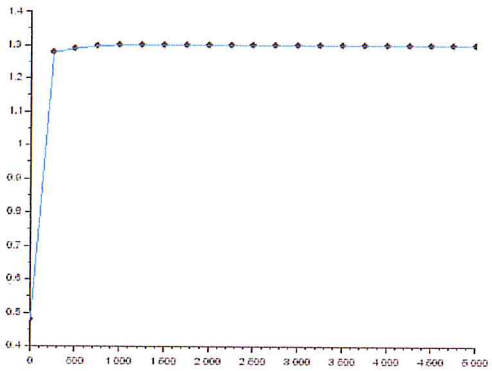


Figure 4.3 : L'hypervolume en fonction du nombre d'itération dans le cas du problème ZDT3

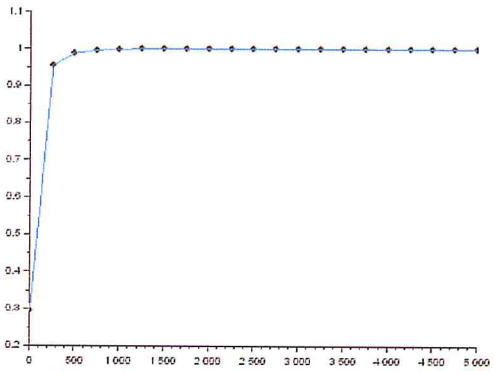


Figure 4.4 : L'hypervolume en fonction du nombre d'itération dans le cas du problème ZDT4

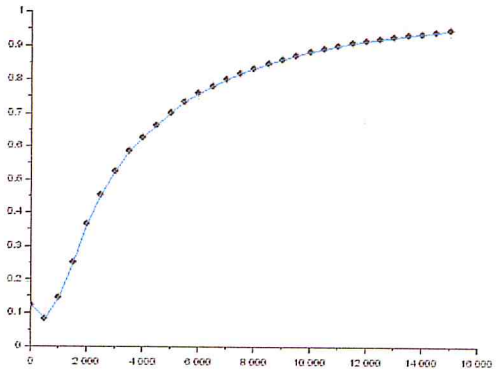


Figure 4.5 : L'hypervolume en fonction du nombre d'itération dans le cas du problème ZDT6

4.4.2 Résultats expérimentaux de l'IGD

Pour évaluer les performances du NSGA-II sur les problèmes à trois objectifs, c'est-dire : DTLZ, nous utilisons une mesure de qualité largement répandue à ce niveau. Il s'agit de la distance générationnelle inversé (IGD), déjà présentée auparavant. En effet si la valeur d'IGD est faible, alors les solutions trouvées par l'algorithme sont proches du front de Pareto. Nous avons l'avons calculé pour chacune des valeur de $n_p = 2, 3$ et pour chacun des 4 problèmes : DTLZ1, DTLZ2, DTLZ4 et DTLZ5. Le tableau ci-dessous donne la valeur de l'IGD pour $M = 3$.

	Génération N_{gt}			**dissmilarité mesure D(p,q)**			**IGD**		
n_p	$n_p = 2$	$n_p = 3$	$n_p = 4$	$n_p = 2$	$n_p = 3$	$n_p = 4$	$n_p = 2$	$n_p = 3$	$n_p = 4$
DTLZ1	244	494	3399	0.0428	0.01	0.0092	3.0629	0.6198	0.27
DTLZ2	194	1521	4404	0.1625	0.1559	0.0797	0.3753	0.3711	0.37
DTLZ4	333	1491	3426	0.1903	0.1679	0.0658	1.0745	0.951	0.3693
DTLZ5	483	885	1499	0.1013	0.0712	0.0151	1.0119	0.887	0.4414

TABLE 4.3 – les valeurs de la mesure de dissmilarité et d'IGD pour $n_p = 2, 3$ et 4.

Pour étudier le comportement de l'algorithme en fonction du nombre d'itérations, nous avons choisi de s'intéresser à la variation de la valeur de l'IGD au cours de l'exécution. Les courbe ci-dessous montrent une allure convergente à partir d'un certain rang.

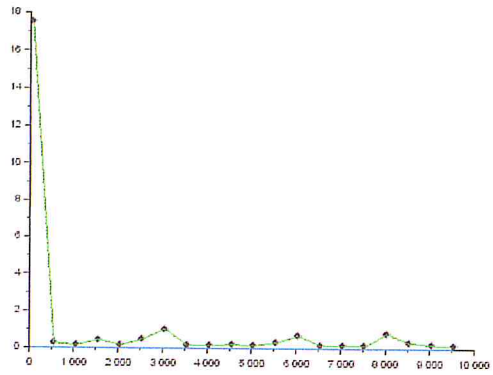


Figure 4.6 : L'IGD en fonction du nombre d'itération du problème DTLZ1 pour 3 objectifs

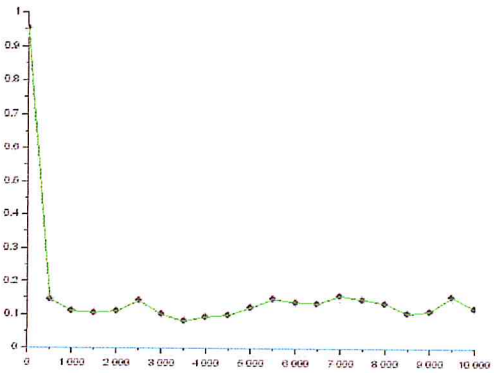


Figure 4.7 : L'IGD en fonction du nombre d'itération du problème DTLZ2 pour 3 objectifs

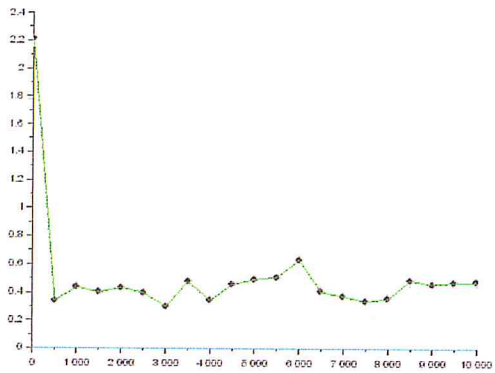


Figure 4.8 : L'IGD en fonction du nombre d'itération du problème DTLZ4 pour 3 objectifs

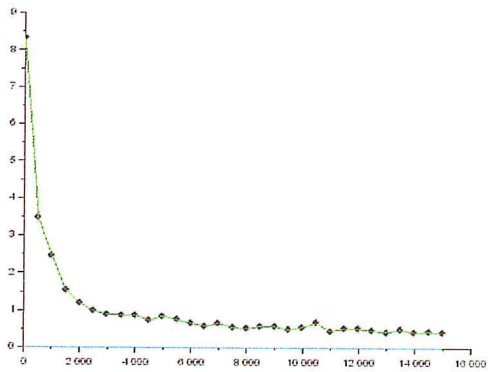


Figure 4.9 : L'IGD en fonction du nombre d'itération du problème DTLZ5 pour 3 objectifs

4.5 Interprétation des Résultats

A travers les résultat de tableau 2 et les figures 4.1-4.5, on peut en conclure que, le NSGA-II doté de l'algorithme2 a pu détecter à quel moment s'arrêter pour les problèmes ZDT1-3 et 6. En effet la valeur de Hypervolume est faible à 3000 génération pour ZDT1-3 et à 7000 générations pour ZDT6. Cela signifié que notre algorithme fonctionne bien pour ces problèmes. Pour ZDT4 la solution optimale n'as pas été approchée.
Pour ce qui est y du tableau 3 et les figures 4.6-7.9. On peut conclure que notre algorithme a bien fonctionné sur les problèmes DTLZ, en effet la valeur d'IGD est faible au moment de l'arrêt.

4.6 Conclusion

Nous avons présenté dans ce chapitre les paramètres utilisés dans nos simulations ainsi que le résultats obtenus sur les différents problèmes tests "ZDT" et "DTLZ".

L'objectif à été d'explorer les performance de l'approche et de voir le comportement de l'algorithme NSGA-II.
La version initiale de NSGA-II donne des résultats qui sont très proche du front de Pareto, en effet, les valeurs de l'hypervolume et de l'IGD obtenues par l'algorithme sont bonnes $n_p = 4$.

Conclusion Générale

Nous avons présenté dans ce mémoire une nouvelle mesure de dissmilarité basée sur l'entropie. Cette mesure nous a permet de construire un critère d'arrêt générique sous forme d'un algorithme permettant de détecter le nombre d'itération adéquat. Cet algorithme pourra être implémenter dans n'importe quel MOEA. Dans notre application, nous avons choisi le NSGA-II et des fonctions testes à deux et trois objectifs. Les appels à l'arrêt de notre algorithme sont chronométrés différemment pour différents problèmes en fonction de leurs niveaux de difficulté, c'est à dire : en termes de la nature des fonctions objectives , de l'espace de recherche (variables), du nombre d'objectifs impliqués et la forme du PF. La précision des appels à l'arrêt est validée par la comparaison de l'approximation du PF en utilisant les métriques de performances des MOEA tels que : l'hypervolume et l'IGD. notre critère pourra être significativement utile pour empêcher des arrêts prématurés pour les MOEA ou un gaspillage de ressources de calcul, une caractéristique critique qui est irrégulièrement restée inchangée.

Parmi les lacunes de ce travail, ont trouve : le nombre d'objectifs considéré car il est nécessaire d'aller au delà de trois objectifs pour voire le comportement de notre algorithme, (Algorithme2),et de le tester sur d'autres MOEA tels que ϵ-MOEA, MOEA-D et d'autres.La méthode d'estimation des distribution de probabilités (P et Q) doit être comparer et tester par rapport à d'autre techniques d'estimation telles que : La méthode de Parzen-Rosenblatt (l'estimation par noyau) ou d'autres méthodes paramétriques et semi paramétrique, car P et Q jouent un rôle de premier plan dans le calcul de la mesure de dissmilarité.

Annexe

A.1 Tests bi-objectif de Zitzler, Deb, Thiele et Tests multi-objectif de Zitzler, Deb.

Dans cet annexe, nous allons présenter les tests bi-objectif (ZDT) et tests multi-objectif (DTLZ), qui ont servi dernièrement de la base commune pour la comparaison des AEMO existants et pour l'évaluation de nouvelles techniques.

A.1.1 Tests bi-objectif de Zitzler, Deb et Thiele

Tous les problèmes se posent sous forme suivante :

$$\min f_1(x_1),$$

$$\min f_2(x) = g(X_2)h(f_1(x_1), g_1(X_2)).$$

où $x = (x_1, X_2)$ et $X_2 = (x_2, ..., x_n)$.

Problème test ZDT1

Le premier de cet ensemble des tests est le plus simple, le front de Pareto correspondant étant continu, convexe et avec la distribution uniforme des solutions long du front.

$$\textbf{ZDT1} = \begin{cases} f_1 = x_1, \\ g = 1 + \dfrac{9}{(n-1)} \sum_{i=2}^{n} x_i, \\ h(x) = 1 - \sqrt{\dfrac{f_1}{g}}, \end{cases}$$

ou $x_i \in \{0, 1\}$ pour tout $i = 1, .., n$ $n = 30$. La solution de ce problème sont que $0 \leq x_i^* \leq 1$ et $x_i^* = 0$ $pour i = 2, ..., n$.

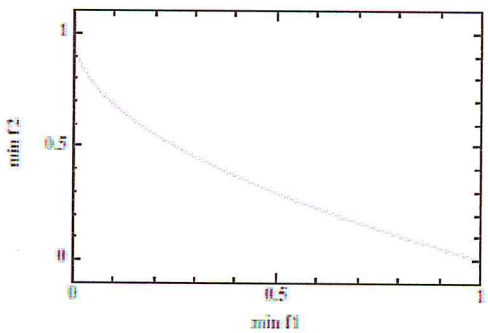


Figure A.1 : SolutiondeZDT1

Problème test ZDT2

La seule difficulté de ce problème consiste en non-convexité du front de Pareto.

$$\mathbf{ZDT2} = \begin{cases} f_1 = x_1, \\ g = 1 + \dfrac{9}{(n-1)} \sum_{i=2}^{n} x_i, \\ h(x) = 1 - \dfrac{f_1}{g}^2, \end{cases}$$

où $x_i \in [0,1]$ pour tout $i = 1,..,n$; $n = 30$. La solution de ce problème sont tels que $0 \leq x_i^* \leq 1$ et $x_i^* = 0$ *pour* $i = 2,...,n$. La front de Pareto du problème ZDT2 est présenté dans la figure A.2.

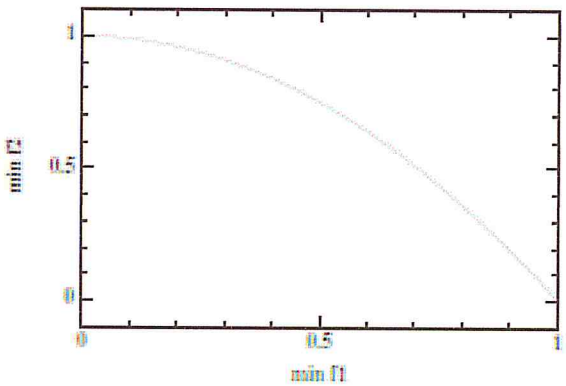


Figure A.2 : Solution de ZDT2

Problème test ZDT3

La difficultè de ce problème c'est que le front de Pareto est discontinu.

$$\mathbf{ZDT3} = \begin{cases} f_1 = x_1, \\ g = 1 + \dfrac{9}{(n-1)} \sum_{i=2}^{n} x_i, \\ h(x) = 1 - \dfrac{f_1}{g} - (\dfrac{f_1}{g}) sin(10 * \pi * f_1), \end{cases}$$

ou $x_i \in [0,1]$ pour tout $i = 1,..,n$;$n = 30$. La solution de ce problème sont telles que $x_i^* = 0$ *pour* $i = 2,...,n$. mais non pas toutes le solution avec $0 \leq x_i^* \leq 1$ sont situè sur le front de Pareto voir figure A.3.

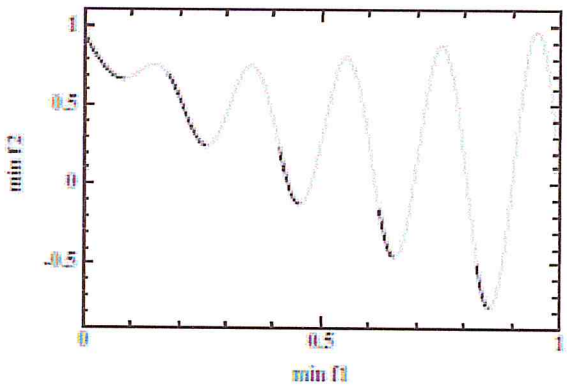


Figure A.3 : Solution de ZDT3

Problème test ZDT4

Ce test modélise la difficulté de la convergence vers le front de Pareto globale à la présence de plusieurs fronts locaux à cause du fait que la fonction g est multi-modale.

$$\mathbf{ZDT4} = \begin{cases} f_1 = x_1, \\ g = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10cos(4)\pi * x_i), \\ h(x) = 1 - \dfrac{f_1}{g}, \end{cases}$$

ou $x_1 \in [0,1]$ et $x_i \in [-5,5]$ pour tout $i = 2,..,n$,$n = 10$. Les optima globaux sont tels que $x_1^* \in \{0,1\}$et $x_i^* = 0$ pour $i = 2,...,n$. Les fronts locaux correspondent à $x_1 \in [0,1]$ et $x_i = 0.5j$, ou j est un entier appartenant$[-10,10]$,$i = 2,..,n$. Quelques fronts locaux et le front global sont présentés dans la figure A.4.

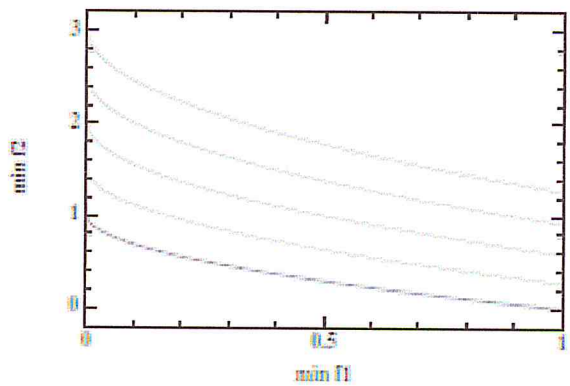


Figure A.4 : Solution de ZDT4

Problème test ZDT6

La particularité de ce problème c'est que les solutions optimales ne sont pas uniformément distribuée le long du front de Pareto. Cette effet est du à la non-linéarité de la fonction f_1.

$$\textbf{ZDT6} = \begin{cases} f_1 = 1 - exp(-4x_1)sin^6(x_i^2 - 10cos(4\pi * x_i)), \\ g = 1 + 9((\sum_{i=2})/9)^{1/4} \\ h(x) = 1 - \sqrt{\frac{f_1}{g}}, \end{cases}$$

ou $x_i \in [0,1]$ pour tout $i = 1,..,n$ $n = 10$. La solution de ce problème sont telles que $0 \leq x_i^* \leq 1$ et $x_i^* = 0$ $pour i = 2,...,n$. La front de Pareto du problème ZDT6 est présenté dans la figure A.5.

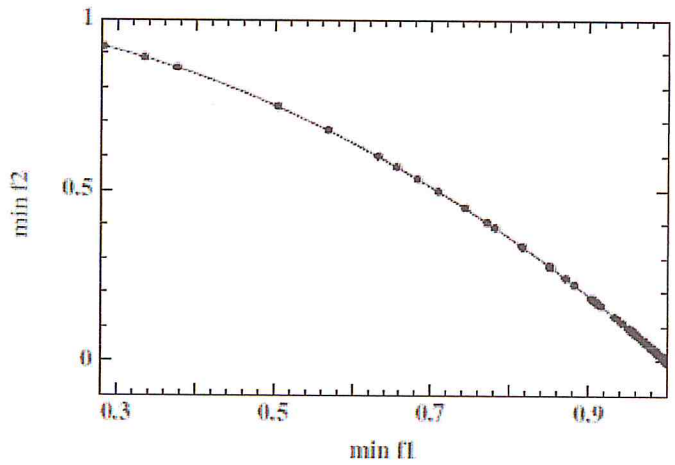


Figure A.5 : Solution de ZDT6

A.1.2 Tests multi-objectif de Deb et Al

Problème test DTLZ1

Comme un problème de test simple, nous construisons un problème de M-objectif avec un front de Pareto optimale linéaire.

$$\textbf{DTLZ1} = \begin{cases} minf_1(x) = \frac{1}{2}x_1x_2...x_{M-1}(1 + g(x_M)), \\ minf_2(x) = \frac{1}{2}x_1x_2...(1 - x_{M-1})(1 + g(x_M)), \\ . \\ . \\ minf_{M-1}(x) = \frac{1}{2}x_1(1 - x_2)(1 + g(x_M)), \\ minf_M(x) = \frac{1}{2}(1 - x_1)(1 + g(x_M)), \end{cases}$$

Les derniére $K = n - M + 1$ variables sont représentés comme des variables de X_M, la fonction $g(X_M)$ exige$|X_M| = k$ variables et doivent toute fonction avec $g \geq 0$ ou :

$$g(X_M) = 100(|X_M| + \sum_{x_i \in X_M} (x_i - 0.5)^2 - cos(20\pi(x_i - 0.5))).$$

Le front de Pareto optimale à $x_i = 0.5$ pour tout $x_i \in X_M$ et se valeurs de fonction objectif sont situés sur l'hyperplan linvaire$\sum_{m=1}^{M} f_m^* = 0.5$; la seule difficulté fournie par ce problème est la convergence dans le front de Pareto optimal.

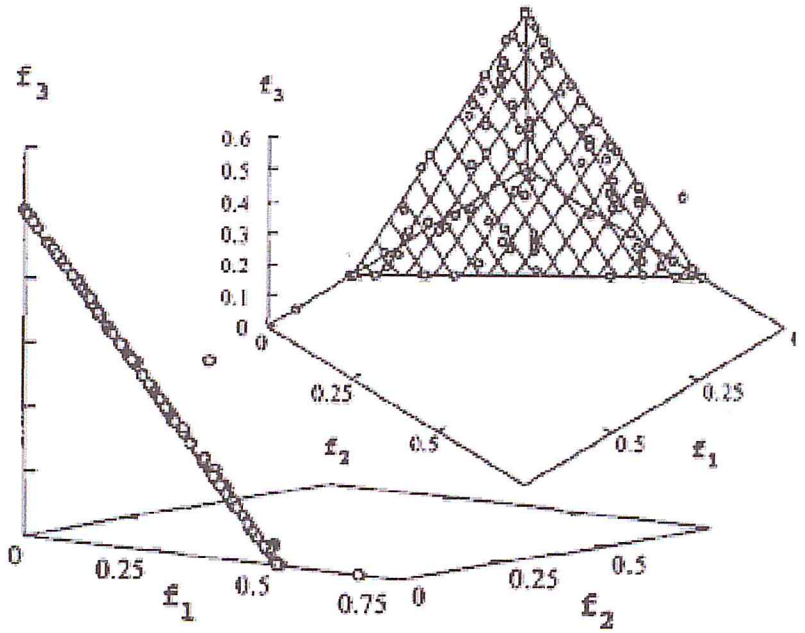


Figure A.6. La population de SPEA2 pour le problème de test DTLZ1 après 300 G. pour $M = 3$.

Problème test DTLZ2

$$\textbf{DTLZ2} = \begin{cases} minf_1(x) = cos(x_1\pi/2)cos(x_2\pi/2)...cos(x_{M-1}\pi/2)(1 + g(x_M)), \\ minf_2(x) = cos(x_1\pi/2)cos(x_2\pi/2)...sin(x_{M-1}\pi/2)(1 + g(x_M)), \\ . \\ . \\ minf_M(x) = sin(x_1\pi/2)(1 + g(x_M)), \\ g(X_M) = \sum_{x_i \in X_M}(x_i - 0.5)^2, \\ 0 \leqslant x_i \leqslant 1, \ pour \ touti = 1, .., n, \end{cases}$$

Le front de Pareto optimal correspond à $x_i = 0.5$ pour tout $x_i \in X_M$ et toutes les valeurs d'objectif doit satisfaire $\sum_{m=1}^{M} f_m^* = 1$.

Problème test DTLZ4

$$\textbf{DTLZ4} = \begin{cases} minf_1(x) = cos(x_1^a\pi/2)cos(x_2^a\pi/2)...cos(x_{M-1}^a\pi/2)(1+g(x_M)), \\ minf_2(x) = cos(x_1^a\pi/2)cos(x_2^a\pi/2)...sin(x_{M-1}^a\pi/2)(1+g(x_M)), \\ . \\ . \\ minf_M(x) = sin(x_1^a\pi/2)(1+g(x_M)), \\ g(X_M) = \sum_{x_i \in X_M}(x_i - 0.5)^2, \\ 0 \leqslant x_i \leqslant 1, \ pour\ touti = 1,..,n, \end{cases}$$

Le paramétre $a = 100$ et Le front de Pareto optimal correspond à $x_i = 0.5$ pour tout $x_i \in X_M$.

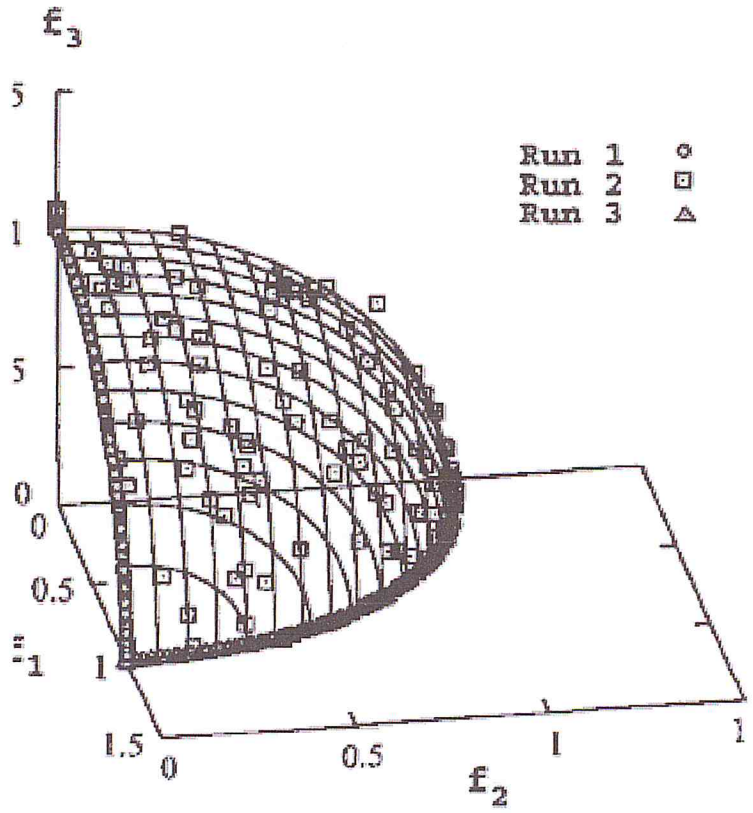


Figure A.7 : La population de SPEA2 pour le problème de test DTLZ4 après 300 G. pour $M = 3$

Problème test DTLZ5

On remplacer O_i par $\pi/4(1+g(r)) * (1+2*g(r)*x_i)$ dans la problème test DTLZ2 pour $i = 2,..,n$ et $g(X_M) = \sum_{x_i \in M_M} x_i^{0.1}$.

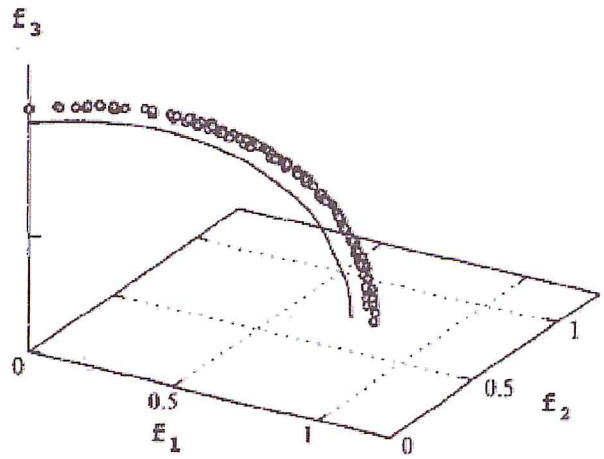


Figure A.8 : La population de SPEA2 pour le problème de test DTLZ5 après 300 G. pour $M = 3$

A.2 Algorithmes

Dans cette partir on proposer l'algorithme d'histogramme multidimensionnel et l'algorithme MOEA Termination détection.

Algorithm 1 Multidimensionnel Histogramme Algorithme for Deux Populations

admission

P : Population réalisable et non dominée correspondant à l'instant t.

Q : Population réalisable et non dominée correspondant à l'instant t +1.

n_b : Nombre de bacs utilisés pour partitionner l'espace de recherche également parmi tous objectif.

résulte

C : Vecteur qui stocke les cellules des régions J_I et P_{NI}

C_q : vecteur qui stocke les cellules de la région Q_{NI}

P_c : vecteur qui stocke le nombre de solutions au sein de chaque cellule de C pour Population P .

Q_c : vecteur qui stocke le nombre de solutions dans chaque cellule de C pour Population Q

Q_{cq} : vecteur qui stocke le nombre de solutions dans chaque cellule de C_q Pour la population Q

1 : Début
2 : pour tout $s \in p$ **faire**
3 : $C \longleftarrow GetCell - id(s, nb)$
4 : si $c \in C$ **alors 5 :**
k={index position de c vecteur C}
6 : $P_{c,k} = P_{c,k} + 1$;
7 : sinon
8 : $C = \{C, C\}$;
9 : $P_c = \{P_c, 1\}$
10 : $Q_c = \{Q_c, 0\}$
11 : finsi
12 : fin pour
13 : pour tout $s \in Q$ **faire**
14 : $C \longleftarrow GetCell - id(s, nb)$
15 : si $c \in C$ **alors**
16 : k={index position de c vecteur C} ;
17 : $Q_{c,k} = Q_{c,k} + 1$;
18 : sinon
19 : si $c \in C_q$ **alors**
20 : k={index position de c vecteur C_q}
21 : $Q_{C_q,k} = Q_{C_q,k} + 1$;
22 : sinon
23 : $C_q = \{C_q, C\}$
24 : $Q_{cq} = \{Q_{cq}, 1\}$
25 : finsi
26 : finsi
27 : fin pour
28 : fin

Algorithm 2 MOEA Termination détection algorithme

admission

n_s : la nombre successive de générations MOEA pour quelle mean et base devition pour la dissmilarité measures .

n_p : le nombre décimal évoluer de quelle mean et base devition pour la dissmilarité mesures.

n_b : nombre de multidimensionnel histogramme.

1 : Début

2 : générer la population P initial

3 : crée Q a partir de la population P .

4 : $(C, Cq, Pc, Qc, Qcq) \longleftarrow MultiHistogramme(P, Q, nb)$

5 : $D_t = 0.$

6 : pour tout $i \in C$ **faire**

7 : $p = Pc, i/|P|$

8 : $q = Qc, i/|Q|$

9 : si $q > 0$ **alors**

10 : $D_t = D_t - [(\frac{P}{2} \log \frac{q}{p}) + (\frac{q}{2} \log \frac{p}{q})]$

11 : sinon if $q = 0$ **then**

12 : $D_t = D_t - p \log q$

13 : finsi

14 : fin pour

15 : pour tout $i \in Cq$ **faire**

16 : $q = Qcq, i/|Q|$

17 : $D_t = D_t - q \log q$

18 : fin pour

19 : Détermine $M_t$etS_t

20 : $\hat{\mathrm{M}}_t = Round(M_t, n_b)$

21 : $\hat{\mathrm{S}}_t = Round(S_t, n_b)$

22 : if $t > n_s$ **alors**

23 : if $[\hat{\mathrm{M}}_t = \hat{\mathrm{M}}_{t-1} = ... = \hat{\mathrm{M}}_{t-n_s}]$ **then** $C_1 = true$

24 : if $[\hat{\mathrm{S}}_t = \hat{\mathrm{S}}_{t-1} = ... = \hat{\mathrm{S}}_{t-n_s}]$ **then** $C_2 = true$

25 : finsi

26 : si $C_1 = true$ et $C_2 = true$ **alors**

27 : Indiquer Q comme population finale et définir $N_{gt} = t$

28 : terminer la course.

29 : sinon

30 : $t = t + 1, C_1 = false, C_2 = false, P = Q$ et retourne étape 3

31 : fin sinon

32 : fin

BIBLIOGRAPHIE

[1] O, Roudenko, *Application des Algorithmes Evolutionnaires aux Problèmes d'Optimisation Multi-Objectif avec Contraintes.*Paris : Ecole polytechnique.2005.

[2] A.Abraham,L,Jain et Goldberg, Evolutionary Multi objectif Optimization-Theeoretical Advances and Applications. Edition Sprinjer.2004.

[3] Souquet Amédeé et Radet Francois-Gérard.Algorithmes Genetiques.2004.

[4] Nicolas.Durand. Algorithmes génétique et autres autils d'optimisation appliqués à la gestion du trafic aérien.

[5] Koussi Koukou : Modélisation et optimisation des transmissions ultra-large Bande à implusi ons Radio dans les Réseaux.

[6] Nicolas Sendrire :Introduction à la théorie de l'information. aout 2007.

[7] Yann Collette-Patrick Siarry : optimisation multi objectif :Edition EYROLLES. 2002.

[8] J .Knowles.L. Thiele et E.Zitzler. A Tutorial on the Performance Assessement of Stochastic Multiobjective Optimizers. TIK repport 214, (TIK), ETH Zurich,2006.16

[9] G.Yen.Z.He, Performance Metric Ensemble for Multi objective Evolutionary Algorithms. IEEE Transaction on Evolutionary computation, vol. 18, NO.1, 2014.

[10] H.Mahideb : l'optimisation multi-objectidf basée sur l'intelligence computationnelle.2014.

[11] M.Haj-Rachid, C. Blooch, W. Ramdhan-Cherif, P. Chationnay, Différentes opérateurs évolutionnaires de permutation :sélections,croisements et mutations.

LIFC.2010.

[12] Dish Kumar Saxena, Arnab Sinha, Joa...o A. Duro, Qingfu Zhang.Entropy-Based Termination Criterion for Multiobjective Evolutionary Algorithms. 2016.

[13] ZHNAN. HE, Performance metrics en sesemble for multiobjecctive evolutionary Algorithms. Beije,g science et technologé. 2011.

[14] Th. Back, D. B. Fogel and Z. Michalewiez. "Handbook of evolutionary computation". Oxford university Press (1997).

[15] H. P. Schwefel. "Nunerical optimization of computer models", john Wiley Sons and Sons, New York, 2nd edition.1995.

[16] R. Koza. Genetic Programming : On the Programming of Computers bymeans of Natural Evolution, MIT Press, Massachussetts,(1999).

[17] I. Rechenberg, Evolutions strategie : Optimierung technischer Systeme und Prinzipien der biologischen Evolution , Frommann-Holzboog, Stuttgart, 1973.

[18] J.H.Holland, "Adaptation in Natural and Artificial Systems". Cambridge, MA : MIT Press. Second edition (1992). First edition, University of Michigan Press, 1975.

[19] M. Ejday, Optimisation Multi-Objectifs à base de Méthamodèle pour les Procédés de Mise en Forme. Doctorat ParisTech. 2011.

[20] E.Alba, B.Dorronsoro, Cellular Genetic Algorithems. Vol 42, Edition Sprinjer. June 2002.