MA-004-172-1

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida

Faculté des Sciences

Département Informatique

مكتبة الكلية

Mémoire

Présenté pour l'obtention du diplôme de Master en Informatique

Option : Ingénierie du logiciel

Thème :

Contribution dans les algorithmes évolutionnaires multi objectifs

Présenté par :

- DIB Abdelatif
- CHABA MOUNA Mustapha

Dirigé par :

Mr. AIT-AKKACHE Mustapha

Soutenu le : 11/10/2013

Devant le jury : - BENSETITI
- OULDAISSA
- BOUGHERARA

Année universitaire : 2012/2013

Remerciements

جامعة سعد دحلب البليدة ★ مكتبة الكلية ★ كلية العلوم

Nous remercions Dieu tout puissant clément et miséricordieux de m'avoir soigné et aidé.

Ensuite nous tenons à remercier notre chers parents pour leur patience et encouragement à finir un chemin commencé il y'a si longtemps.

Nous tenons, avant tout, à exprimer nos profonde gratitude à monsieur AIT-AKKACHE Mustapha, professeur à l'université de BLIDA, qui a assumé la direction de ce travail. Qu'il veuille bien trouver ici l'expression de ma reconnaissance pour son dévouement, sa patience, sa disponibilité, ses conseils et son aide constant qu'il m'a apporté tout au long de ce travail.

Je remercie les membres de jury qui ont accepté de juger ce travail.

J'adresse mes vifs remerciements à tous les enseignants qui, par leur enseignement, leur encouragement et leur aide, ont contribués à ma formation durant toutes mes études.

J'exprime mon remerciement à tous ceux qui ont participé au bon déroulement de ce travail.

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 sœurs, 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 2012-2013 de post-graduation.

A tous ceux qui aiment Abdelatif et ceux qui Abdelatif aime.

Abdelatif

Dédicace

Je dédie ce modeste travail à mes très chers et aimables parents C.Abdlkader, B.Rokia qui sont toute ma fierté et à qui Je dois tout ce que je suis aujourd'hui et, à ceux que j'aime tant et qui sont ma joie de vivre qu'ALLAH vous protège et ne me privera jamais de vous.

A mes chères sœurs Iman, Hayat,Siham,Khadidja, Manal.
aux petits Hadil ,Abdellah,Loudjayn
pour qui me donne le bonheur la joie et l'harmonie Nadjah

A tous ce qui m'aime et à tous qui j'aime

Mustapha

Résumé

Le but du travail est d'étudier la possibilité de ramener plus d'efficacité dans les algorithmes évolutionnaires multi objectifs à travers Incorporation de connaissances sur le vrai front Pareto. Ces informations se résument dans un nombre très réduit de solutions médianes se trouvant sur le vrai front et qui sont déterminées grâce à un algorithme génétique bien définie et dont l'appellation est LMOGA (Lexico-Max-Ordering Genetic Algorithm) [7].

Le travail consiste dans un premier temps d'incorporer dans les différents algorithmes évolutionnaires [35] considérées les connaissances fournées par LMOGA ensuite de faire une série de simulations en utilisant les fonctions testes de Zitzler, Deb &Thiele [36]. Ces simulations permettent de comparer les algorithmes résultants par rapports à leurs ainés (telles que SPEA2, NSGAII) en termes de qualité de solutions obtenues et de vitesse de convergence.

Mots clés

Optimisation multi-objectives, Dominance, solution de Pareto, Algorithmes évolutionnaires.

Introduction

Abstract

The aim of this work is to study the possibility of bringing more efficiency in multi-objective evolutionary algorithms through Incorporation of knowledge about the true Pareto Front. This information can be summarized in a very small medians solutions found on the right edge and are determined using a well-defined genetic algorithm number and whose name is LMOGA (Lexico-Max-Ordering Genetic Algorithm) [7].

The work in the first step is to incorporate into different evolutionary algorithms [35] considered the knowledge batches by LMOGA then make a series of simulations using the tested functions Zitzler, Deb & Thiele [36]. These simulations compare the resulting algorithms with regard to their elders (such as NSGAII, SPEA2) in terms of quality of solutions obtained and convergence speed.

Abréviation

Abréviation	Signification
AE	Algorithmes Evolutionnaires.
MOEAs	Multi-objectifs Evolutionary Algorithms
EMOA	Evolutionary Multi-objectifs Algorithm
NSGAII	Non Dominated Sorting Genetic Algorithm II
SPEA-II	Strength Pareto Evolutionary Algorithm II
LMOGA	Lexico-Max-Ordering Gentic Algorithm
LMO	Lexico-Max-Ordering
ER	Error Ration
GD	Generational Distance
SP	Spacing
SCM	Set Coverage Metric

Liste des Figures

Liste des Tableaux

SOMMAIRE

CHAPITRE II :

Les algorithmes évolutionnaires multi-objectifs

CHAPITRE III :

L'algorithme LMOGA et problème test

CHAPITRE IV :

Simulation et comparaison

Introduction générale

Introduction générale

De très nombreux problèmes du monde réel impliquent l'optimisation simultanée de plusieurs critères qui sont en général contradictoires. Dans notre économie, par exemple, le profit et les ventes doivent être maximisés et les coûts devraient être aussi bas que possible. La notion de solution optimale unique disparait pour des problèmes de ce type au profit de la notion de compromis entre divers objectifs. Ces compromis donnent naissance à une multitude de solutions dite solutions *de Pareto* et l'ensemble de solutions que l'on obtient à la fin de la recherche est la *surface de compromis*. La recherche de ces solutions de Pareto passe par la conception d'algorithmes différents de ceux destinés aux problèmes d'optimisation mono-objectif.

Durant ces dernières décennies, les Algorithmes Evolutionnaires Multi-objectifs (MOEAs) ont obtenues une attention particulière parmi les chercheurs et les utilisateurs de l'optimisation multi-objective puisqu'ils travaillent sur une population de solutions candidats, à la différence de la plupart des méthodes traditionnelles qui manipulent en général un unique point de l'espace de recherche. Cette particularité des Algorithmes Evolutionnaires (AE) rend possible l'obtention d'un ensemble de Pareto approché en un seul essai de l'algorithme.

Les différentes recherches et études ont démontré que les algorithmes évolutionnaires Multi-objectifs [36] peuvent progresser de différentes manières vers l'ensemble Pareto-optimal, mais malgré cela aucun de ces algorithmes évolutionnaires multi-objectifs (MOEAs) ne peut garantir la convergence vers les véritables solutions Pareto optimales. Cela est dû principalement aux informations utilisées par la procédure de recherche et le test d'arrêt qui se limite aux nombre d'itération fixe au préalable. Ces informations sont relatives et concernent le véritable front de Pareto du problème à résoudre et peuvent généralement être incorporées soit lors de l'initialisation de l'algorithme ou soit par les opérateurs de variation au cours de l'exécution de l'algorithme.

Le but du travail est d'évaluer l'efficacité provoquée dans un algorithme évolutionnaire multi objectifs par l'incorporation de connaissances sur le vrai front de Pareto lors du processus d'initialisation des l'Algorithmes. Ces connaissances, dans notre cas, est un unique point (solution) appartenant au vrai front. Cette solution est obtenue, dans notre cas, grâce à un autre type d'algorithme évolutionnaire [36] qui permet la recherche d'un genre particulier de solutions dites LMO (relativement à une dérivée de la dominance de Pareto dite la dominance Lexico-Max_Ordering [7]. Le travail consiste en un ensemble d'expérimentations, on s'intéresse dans un premier temps à l'implantation des algorithmes évolutionnaires multi-objectifs tel que SPEA2, NSGAII et LMOGA. Ensuite de faire une série de simulations avec et sans l'incorporation des connaissances sur les fonctions testes de Zitzler, Deb &Thiele [1] [18] [24] [38]. Ces simulations permettent de faire les comparaisons en termes de qualité de solutions obtenues et de vitesse de convergence.

L'organisation de ce mémoire est la suivante, Le premier chapitre est consacré à la présentation de l'optimisation multi-objectifs, et des concepts fondamentaux tels que la dominance de Pareto et ces dérivées. Dans le deuxième chapitre on trouve un état de l'art sur les algorithmes évolutionnaires multi objectif (EMOA) et en particulier SPEA2 et NSGAII. Dans le troisième chapitre nous présentons l'algorithme LMOGA qui nous permet d'obtenir les solutions LMO, les fonctions testes sur les quelles nous appliquant nous algorithmes ainsi que les métriques utiles pour mesurer et comparer de nos résultats obtenus par nos algorithmes. Le dernier chapitre est entièrement dédié aux expérimentations numériques et leurs comparaisons.

Chapitre I

Optimisation multi-objectifs

I.1. Introduction

Dans ce chapitre, nous présentons les principes de base de l'optimisation multi objectif. Dans un premier temps, nous rappelons quelques notions élémentaires de l'optimisation multi-objectif, tels que les notions de dominance et de l'optimalité de Pareto.

Nous donnerons aussi une classification des problèmes d'optimisation multi objectif. Puis on présentera quelques méthodes.et On finira par définir quelques approches de résolution pour traiter un problème multi-objectif.

Dans la plus part des problèmes pratiques d'optimisation, plusieurs critères sont à prendre en considération afin d'obtenir une solution satisfaisante. Comme son nom l'indique, l'optimisation multi-objectif a pour but d'optimiser plusieurs objectifs simultanément. Ces objectifs sont dans le cas général en conflit : l'amélioration d'un objectif provoque la détérioration d'un autre objectif. Par conséquent, le résultat final de l'optimisation n'est plus donné par une solution unique mais plutôt par un ensemble de solutions qui représentent chacune un compromis entre les différents objectifs à optimiser. [37]

I.2.Problème d'optimisation multi-Objectif

Un problème d'optimisation multi-objectif est un problème qui possède plusieurs fonctions objectif qui sont à minimiser ou à maximiser et un certain nombre de contraintes à satisfaire. La forme générale d'un problème d'optimisation multi-objectif est donnée par le système d'équations suivant :

$$\begin{cases} Minimiser/\ Maximiser f_m(X)\ m = 1,2,\dots,M; \\ g_j(x) \geq 0, \qquad j = 1,2,\dots,J; \\ h_k(x) = 0, \qquad k = 1,2,\dots,K; \\ x_i^{(L)} \leq x_i \leq x_i^{(U)}, \qquad i = 1,2,\dots,n; \end{cases} \tag{I.1}$$

Le vecteur x est un vecteur de n variables de décision

$$x = (x_1, x_2, \dots . x_n)^T \tag{I.2}$$

$.x_i^{(L)} et x_i^{(U)}$ sont les bornes inférieure et supérieure de la variable x_1, respectivement. Ces variables définissent l'espace de décision ou l'espace de recherche D.

Généralement, un élément de l'espace de recherche est appelé solution possible ou potentielle.

Les termes $g_j(x)$ et $h_k(x)$ sont les fonctions contraintes. Les contraintes d'inégalité sont traitées en tant que contraintes de type "supérieur ou égal" étant donné qu'elles contraintes de type "inférieur ou égal" peuvent être traitées par dualité. Une solution x qui ne satisfait pas la totalité des (J + K) contraintes est dite solution infaisable. L'ensemble des solutions faisables constitue la région faisable (ou réalisable) S($S \subseteq D$).

$$\text{Le vecteur } f(x) = (f_1(x), f_2(x), \dots \dots f_M(x))^T \qquad (I.3)$$

Est le vecteur objectif. Chacune des M fonctions objectives est à maximiser ou à minimiser selon le problème traité. En utilisant le principe de dualité [26], un problème de maximisation peut être ramené à un problème de minimisation en multipliant la fonction objective par -1.
Dans le reste de ce document, nous supposons que toutes les fonctions objectives sont à minimiser. [37]

Sauf mention contraire explicite, tous les énoncés et définitions seront donnés dans le cadre de problèmes de minimisation. En effet un problème de maximisation peut être aisément transformé en problème de minimisation en considérant l'équivalence suivante :

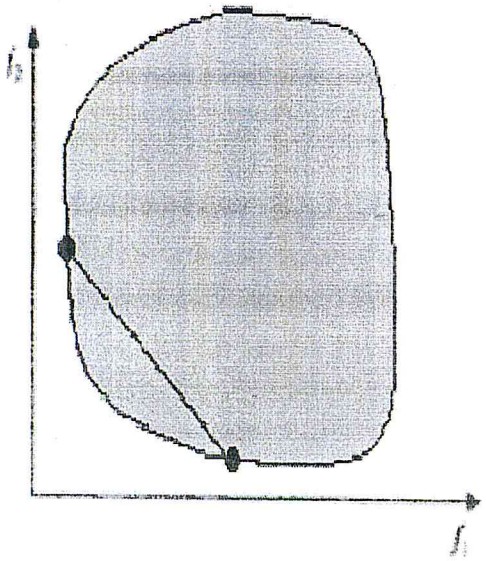
$$Maximiser f(\vec{x}) \Leftrightarrow Minimiser - f(\vec{x}) \qquad (I.4)$$

I.3.La convexité d'un espace de recherche

Nous citons ici la définition liée à la convexité car cette notion est utile et elle concerne l'ensemble de solutions réalisables S. S est dit convexe si et seulement si : étant donnés deux point A et B quelconques de cette ensemble S, l'ensemble des points reliant A à B appartient à S autrement :

$$\forall A et B \in S,\ segment(A,B) \subset S \qquad (I.5)$$

La convexité est le premier indicateur de la difficulté du problème. En effet, certaines méthodes d'optimisation trouvent des difficultés dans la résolution des problèmes non convexes de manière optimale [41].



FigureI.1: Un espace de recherche Convexe

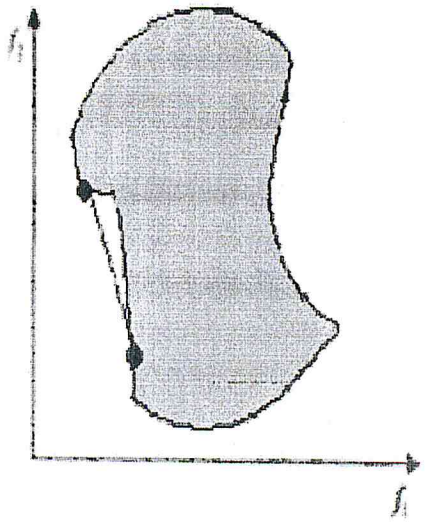


Figure I.2: Un espace non convexe [38]

I.4. Dominance et Optimalité de Pareto

Lorsque l'on cherche à résoudre un problème d'optimisation multi objectifs donné, la plupart du temps, on trouve une multitude de solutions du fait que certaines des objectifs sont contradictoires. Le concept d'optimalité, qui nous permettra de définir les solutions obtenues, est le compromis. Pour identifier ces meilleurs compromis on doit définir une relation d'ordre entre ces éléments. Ces relations d'ordre pour le cas du multi objectif sont appelées relations de dominance. [41]

I.4.1.La dominance au sens de Pareto

Elle est la plus célèbre et la plus utilisée des dominances. AuXIX^{ème} siècle, Vilfredo Pareto, un mathématicien italien, formule le concept : « dans un problème multi objectif, il existe un équilibre tel que l'on ne peut pas améliorer un critère sans détériorer au moins un des autres critères ».

De manière à définir clairement et formellement cette notion, les relations =, ≤ et < usuelles sont étendues aux vecteurs. [41]

Définition :(La dominance au sens de Pareto)

Soient,$\vec{u} = (u_1, u_2, \ldots u_m)$ et $\vec{v} = (v_1, v_2, \ldots v_m)$, deux solutions faisablespour problème de minimisation (I.1) :

- On dit que la solution $\vec{u}$ domine $\vec{v}$, (qu'ontnote $\vec{u} \succ \vec{v}$), ou encore $\vec{v}$est dominé par$\vec{u}$, (qu'on note$\vec{v} \prec \vec{u}$),si et seulement si $\vec{f}(\vec{u}) < \vec{f}(\vec{v})$. (I.6)

- On dit que la solution $\vec{u}$ domine faiblement $\vec{v}$, (qu'on note $\vec{u} \succcurlyeq \vec{v}$), ou encore $\vec{v}$ est dominé faiblement par $\vec{u}$, (qu'on note $\vec{v} \preccurlyeq \vec{u}$), si et seulement si $\vec{f}(\vec{u}) \leq \vec{f}(\vec{v})$. (I.7)

- On dit que la solution $\vec{u}$ non domine $\vec{v}$, (qu'on note $\vec{u} \sim \vec{v}$), si et seulement si $not\vec{f}(\vec{u}) \leq (\vec{f}(\vec{v}) \wedge \text{not } \vec{f}(\vec{v}) \leq (\vec{f}(\vec{u})$. (I.8)

Remarque :Pour le problème de maximisation (I.1), il suffit de remplacer dans les définitions ci-dessus les symboles $<$ ($respectivement \leq$)par le symbole$>$ ($respectivementt \geq$).

Les solutions qui dominent les autres et qui ne se dominent pas entre elles sont appelées solutions optimales au sens de Pareto (ou solutions non dominées). On distingue deux types d'optimalité : l'optimalité locale et l'optimalité globale au sens de Pareto *(Figure I.3)*. [41]

I.4.1.1Optimalité locale au sens de Pareto

Un vecteur $\vec{x} \in R^n$ 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 $\vec{x}' \in R^n \cap B(\vec{x}, \delta)$, où $B(\vec{x}, \delta)$ représente une boule de centre $\vec{x}$ et de rayon δ. [42]

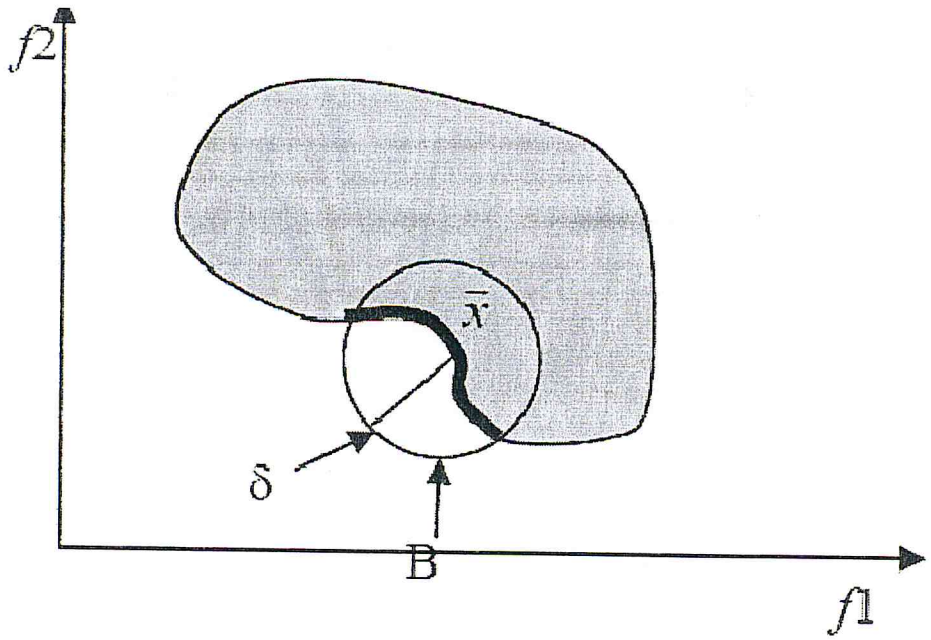


Figure I.3: L'optimalité locale au sens de Pareto

I.4.1.2Optimalité 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$\vec{x}'$.

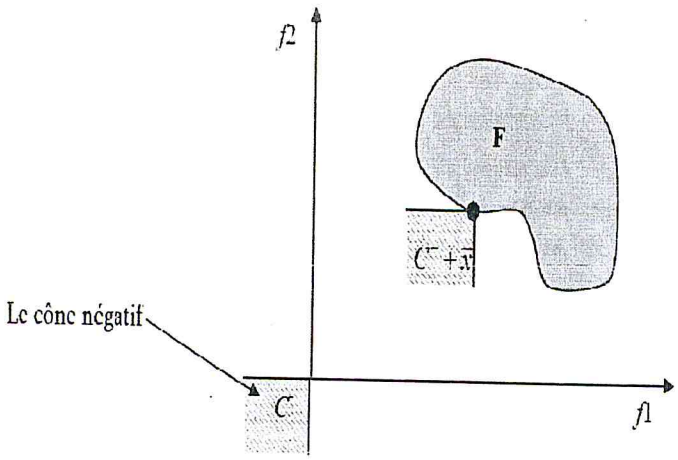
La différence entre cette définition et celle de l'optimalité locale tient dans le fait que l'on ne considère plus une restriction de l'ensembleR^n. [42]

Le 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 :

$$(\overline{C} + \vec{x}) \cap F = \{\vec{x}\}$$

Où F désigne l'espace de solutions réalisables *(figure I.4)*. [42]



FigureI.4 : Le théorème de contact

Définition : (Le cône négatif)

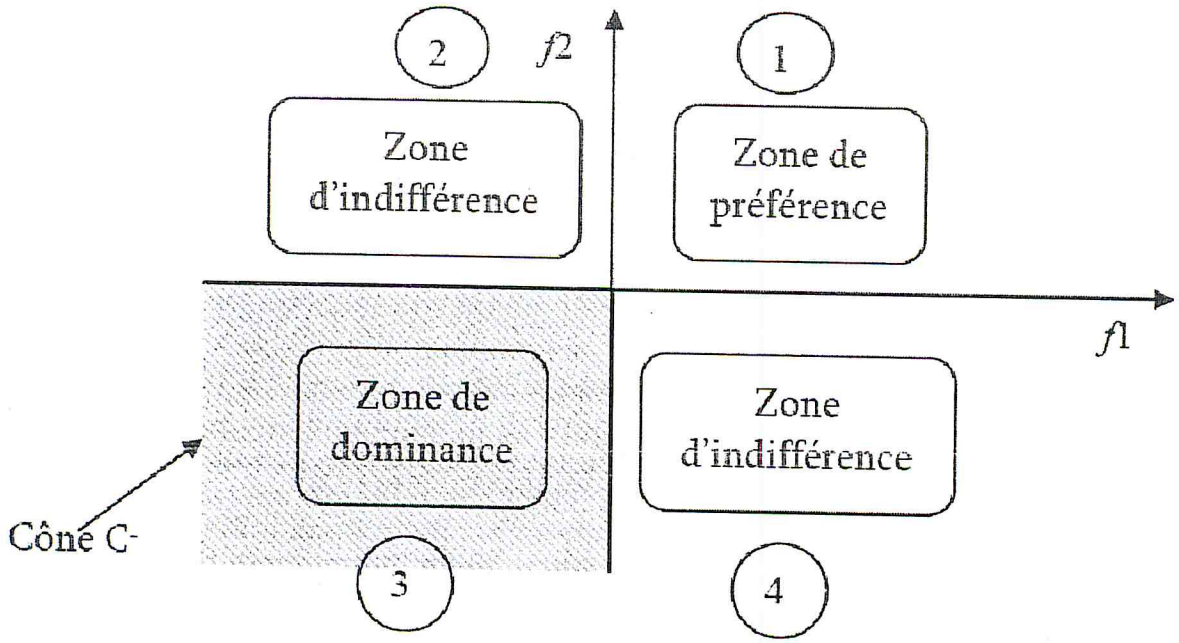
Un cône négatif est définit dans R^k de la manière suivante :

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

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 I.5)*. Cette figure reprend le découpage défini par le cône négatif que l'on a introduit précédemment et l'étend à tout l'espace. [42]

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. [42]



FigureI.5 : Les niveaux de préférence dans la relation de dominance.

I.4.2. frontière de Pareto (surface de compromis)

La frontière, appelée aussi le front de Pareto, est l'ensemble des points Pareto optimaux. La *figure I.6* présente deux formes, parmi une multitude de fronts de Pareto.

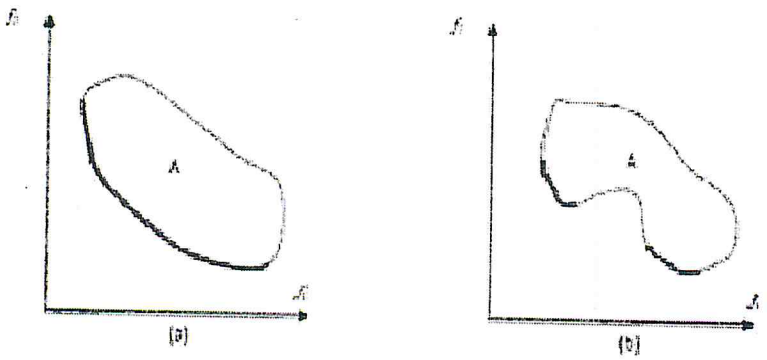


Figure I.6: exemple des fronts de Pareto. A : domaine a réalisé (bi-objectifs) [3].

I.4.2.1 La représentation de la surface de compromis

Toutes les représentations de la surface de compromis, pour un même problème, ne sont pas équivalentes. En effet, la représentation idéale de la surface de compromis devra être constituée de points solution de notre problème répartis de manière uniforme sur la surface de compromis *(voire figure I.7)*.

Par exemple dans la figure 1.7. les points représentant la surface de compromis ne sont pas répartis de manière uniforme. L'utilisateur n'aura alors pas en sa possession un ensemble de solutions très utile. En effet, s'il décide que la solution qu'il avait choisie ne lui convient pas,le choix d'une autre solution risque de 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.

La détermination d'une bonne représentation de la surface de compromis sera un critère de choix d'une méthode d'optimisation multi-objectif [42].

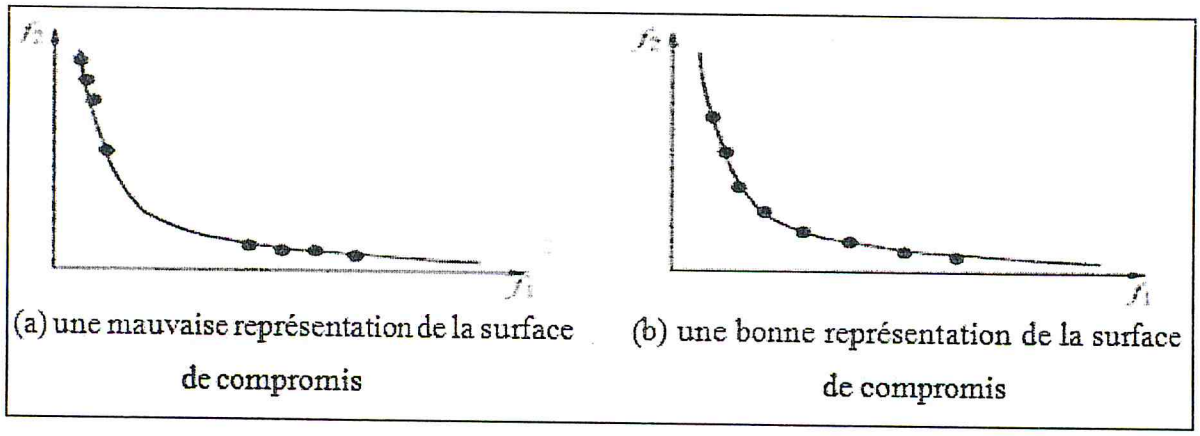


Figure I .7 : la représentation de la surface de compromis

I.4.2.2Vecteur idéal et vecteur Nadir

Le vecteur idéalz^*du problème (I.1) est le vecteur de l'espace des critères dont chaque composante $z_m$est la solution du problème de minimisation de la fonction $f_m$sous les contraintes des problèmes (I.1) *(figure I.8)*.Généralement, ce vecteur ne correspond pas à une solution de l'espace de décision mais il est quelques fois utile en tant qu'une référence, par exemple, lors de la normalisation des valeurs des objectifs [36].

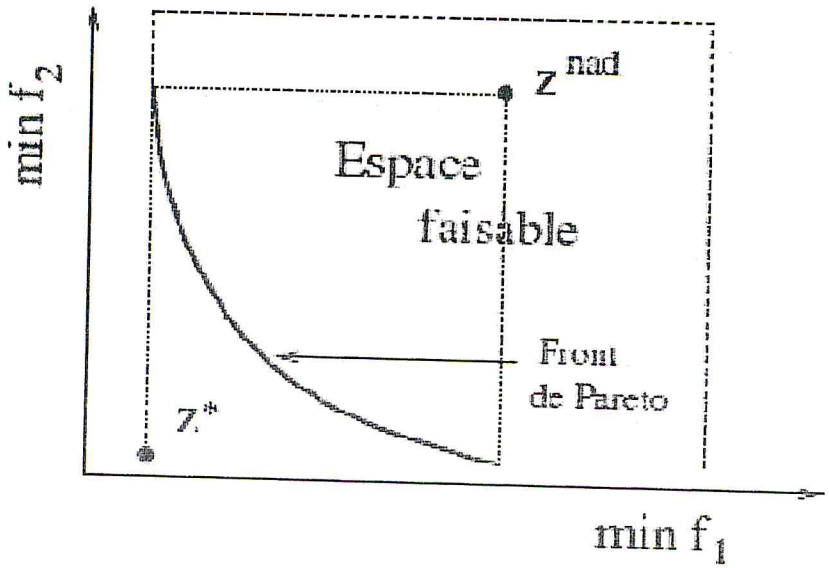


Figure I.8 : Le vecteur idéal z^* et vecteur de Nadir z^{nad} [36]

A la différence du vecteur idéal qui représente les bornes inferieures de chaque objectif dans l'espace faisable, le vecteur idéal de Nadir z^{nad} correspond à leurs bornes supérieures sur la surface de Pareto et non pas dans tout l'espace faisable *(figure I.8)*. Ce vecteur est bien plus difficile de trouver que le vecteur idéal.

Pour certains problèmes, la méthode de payofftable peut être utilisée. Le vecteur idéal de Nadir peut correspond à une ou à aucun solution existante, en fonction des problèmes (notamment, de la convexité du domaine de recherche) [36].

Pour normaliser chaque objectif, le vecteur idéal et le vecteur Nadir sont en particulier utilisée de façon suivante :

$$f_m^{norm} = \frac{f_m - z_m^*}{z_m^{nad} - z_m^*} \quad (I.9)$$

I.5. Les relations dérivées de la dominance

La relation de dominance n'offre pas de degrés de liberté dans sa définition. Par exemple, il n'est pas possible d'inclure dans la définition de la relation de dominance de Pareto une préférence d'un objectif par rapport à un autre. C'est pour contrecarrer ce manque de flexibilité que des relations dérivées de la relation de dominance ont été développées [42].

I.5.1. L'optimisation lexicographie

Cette définition de l'optimalité permet d'inclure une préférence entre objectifs

Définition : optimalité au sens lexicographie

Une solution $\overrightarrow{x^*} \in R^n$ est optimale au sens lexicographique si :

$$\overrightarrow{x^*} \succcurlyeq_{lex} \vec{x}. \forall \vec{x} \in R^n - \{\overrightarrow{x^*}\} \qquad \text{(I.10)}$$

Si $\vec{x}, \vec{y} \in R^n$, on dit que $\vec{x} \succcurlyeq_{lex} \vec{y}$ s'il existe une valeur d'index q^* telle que $f_q(\vec{x}) = f_q(\vec{y})$ pour $q = 1, \ldots, (q^* - 1)$ et $f_q^*(\vec{x}) < f_q^*(\vec{y})$. les relations entre $f_q(\vec{x})$ et $f_q(\vec{y})$ pour $q \geq q^*$ ne sont pas prises en compte car nous nous arrêtons a l'indice q^* (c'est le premier indice pour lequel $f_q(\vec{x}) < f_q(\vec{y})$). Cette comparaison lexicographique des objectifs est noté par :

$$\left(f(\vec{x}) \leq_{lex} f(\vec{y})\right)$$

I.5.2. L'optimisation max-ordering (optimalité maximale)

Cette relation, contrairement aux précédentes, ne permet pas d'introduire une préférence entre objectifs [31].

Définition : optimalité maximale

Une solution $\vec{x} \in R^n$ est max-ordering optimale si la valeur du pire(le max) objectif est aussi petite que possible, c'est à dire :

$$Max\, f_q(\vec{x}) \leq Max f_j(\vec{y})$$
$$q \in \{1, \ldots, n\} \quad j \in \{1, \ldots, n\} \text{ et } \vec{y} \in R^n$$

I.5.3. Optimalitélexico-max-ordering

Ce type d'optimisation combine les caractéristiques de Pareto, max-ordering et l'optimalité lexicographique.

Définition1 : Pour tout élément $\vec{x} \in R^n$ on définit $Sort\,(\vec{x}) = (Sort_1(\vec{x}), \ldots, Sort_n(\vec{x}))$ comme étant le vecteur contenant les composantes de $\vec{x}$ dans un ordre décroissant, c'est-à-dire : $Sort_1(\vec{x}) \geq \cdots \geq Sort_n(\vec{x})$

Définition 2 **:**

Une solution faisable $\vec{x} \in R^n$ est dite solution lexicographique max-ordering (Lex-MO) si le vecteur des objectifs associe à $\vec{x}$ est lexicographiquement minimal par rapport à $Sort\,(f(\vec{x}))$ où $(f = f_1, f_2, \ldots, f_n)$ représente des fonctions objectives.

Donc :

$$Sort(f(x)) \leq_{lex} sort(f(x)). x' \in R^n \quad \text{(I.11)}$$

Maintenant nous présentons un simple exemple qui illustre la relation entre la solution de LEX-MO et de Pareto.

Exemple :

Considérons un problème dont l'ensemble réalisable de la solution est F = {a, b, c, d, e}

Supposons que les valeurs de la fonction objective et les vecteurs Sort sont présentés dans le *(Tableau I.1)*.

Tableau I.1 : les valeurs des objectifs et de Sort [31]

F	F(X)	Sort (F(X))
a	(1, 3, 8, 2, 4)	(8, 4, 3, 2, 1)
b	(4, 3, 8, 1, 1)	(8, 4, 3, 1, 1)
c	(7, 5, 4, 6, 1)	(7, 6, 5, 4, 1)
d	(3, 7, 4, 6, 5)	(7, 6, 5, 4, 3)
e	(4, 7, 5, 6, 5)	(7, 6, 5, 5, 4)

Il est à noter que a, b, c et d sont des solutions de Pareto .La solution optimale lexico graphiquement est évidemment a l'ensemble de solution de max-ordering est c, d, e. Alor que c est la solution unique de LEX-Mo ainsi est une solution de Pareto et max-ordering optimale [31].

I.6. Approches de résolution de problème Multi-objectif

Un grand nombre d'approches existent pour résoudre les problèmes multi-objectifs [41] Certains utilisent des connaissances du problème pour fixer des préférences sur les critères et ainsi contourner l'aspect multicritère du problème. D'autres mettent tous les critères au même niveau d'importance, mais là aussi il existe plusieurs façons de réaliser une telle opération [15].Ci –dessous (figure I.9) un schéma qui représente tous les approches de résolution de problème multi-objectif.

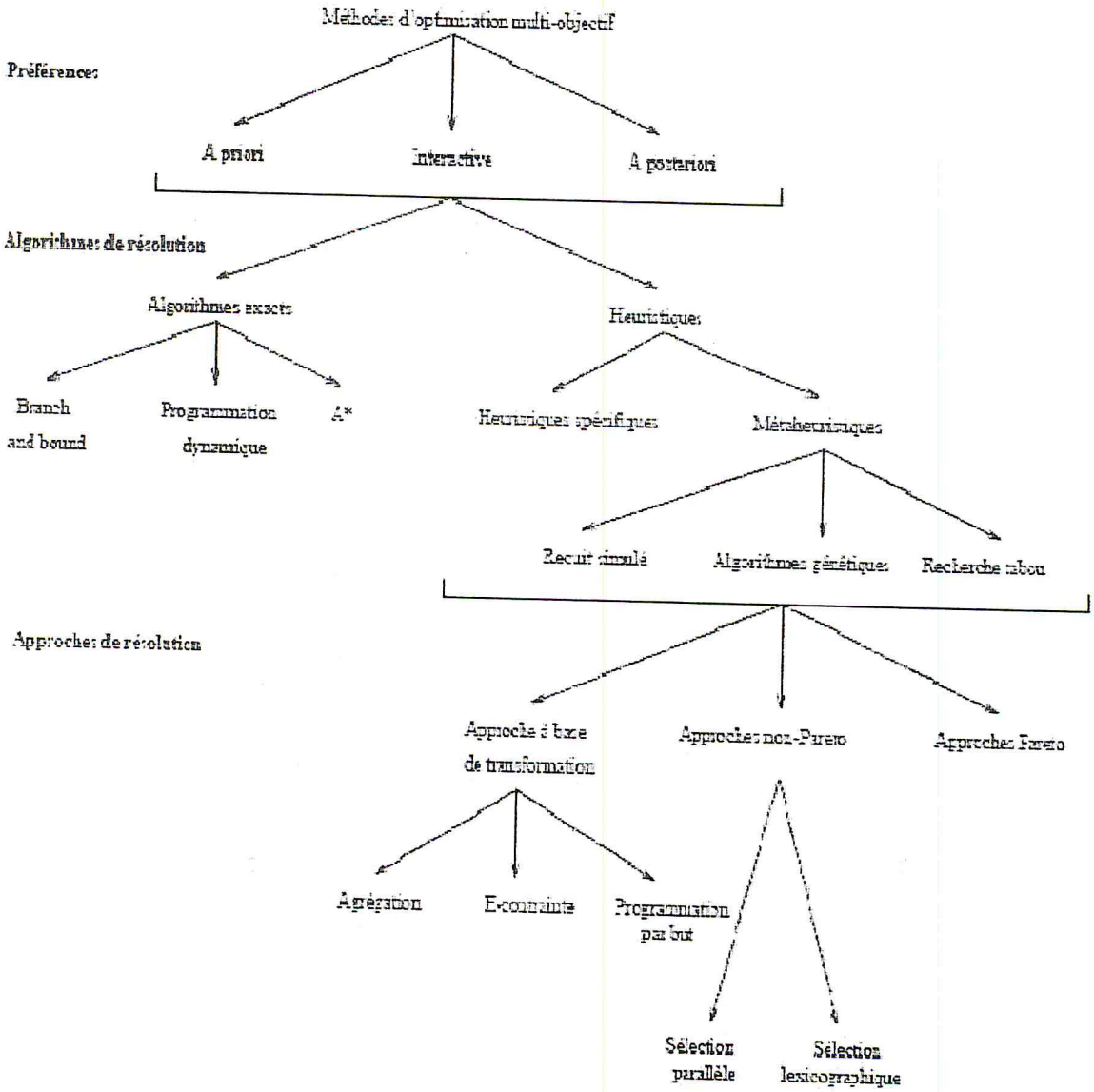


Figure I.9 : Classification des approches MOO. [15]

I.6.1. Les Approches de résolutions à base de transformation

Ces approches transforment le problème initial afin de se ramener à la résolution d'un ou plusieurs problèmes mono-objectif. Parmi ces méthodes, on peut citer les méthodes d'agrégation, les méthodesε-contraintesou encore les méthodes de programmation par but.

En général ces méthodes nécessitent une connaissance du problème et ne fournissent qu'une seule solution [8]. Elles peuvent alors être classées dans la famille des méthodes d'optimisation a priori.

I.6.2. Les Approches Non Pareto

Ces approches sont ni Pareto ni agrégatives. Elles transforment le problème d'origine et elles effectuent leur recherche en traitant indépendamment chacun des objectifs et d'une façon équivalente [42]. On trouve parmi ces approches :

❖ **Les Approches à sélection parallèle**

L'exemple le plus classique est l'algorithme VEGA (VectorEvaluatedGeneticalgorithm) [20]. Cette méthode tente d'optimiser en parallèle chaque objectif sur un ensemble de solutions indépendamment des autres objectifs. Elle produit en réalité des solutions expertes pour un seul objectif. Ce type de méthodes souvent du mal à trouver les solutions de compromis puisqu'elles se focalisent sur les parties extrêmes du front.

❖ **Les Approches à sélection lexicographique ou préférences**

Dans cette méthode, les objectifs sont considérés rangés par ordre d'importance par le décideur [42].La solution optimale est alors obtenue en minimisant le premier objectif d'abord, le deuxième et ainsi de suite jusqu'au dernier. Ce type de méthode représente les mêmes inconvénients que les précédentes.

I.6.3. Les Approches Pareto

Ces approches utilisent la notion de dominance pour comparer les solutions entre elles. L'un des premiers à discuter de l'intérêt de l'utilisation de la notion de dominance pour la recherche de solutions est [11]. Ce type de méthodes ne fait subir aucune transformation au problème multi-objectif ni une préférence pour un objectif par rapport à un autre.

Les objectifs sont traités de la même façon et les solutions optimales sont celles qui ne sont pas dominées au sens de Pareto. En général, dans ce type d'approche, une seule exécution suffit pour approcher la frontière Pareto. En plus d'après [9] même si la nature contradictoire des critères n'est pas prouvée, les métaheuristiques basées sur la dominance de Pareto pourraient trouver la solution idéale et c'est la meilleure pour tous les critères.

I.7. Conclusion

Dans ce chapitre nous avons entamé l'état de l'art sur l'optimisation multi-objectifs, de telle sorte qu'on à passer par la définition des problèmes d'optimisation multi-objectifs, ses contraintes, ses propriétés, l'aspect de surface de compromis Approches de résolution de problème Multi-objectif(Pareto et non Pareto).pour résoudre les problèmes d'optimisation multi-objectifs on trouve les algorithmes évolutionnaires qui exercent le principe de la génétiques et qui ils ont démontrées leur efficacité et qu'on va les voir dans le chapitre suivant.

Chapitre II

Les algorithmes évolutionnaires multi-objectifs

II.1. Introduction

La théorie évolutive darwinienne, proposée par Charles Darwin, repose sur deux postulats simples:

- Dans chaque environnement, seul les espèces les mieux adaptées survivent au cours du temps, les autres sont condamnées à disparaître.

- Au sein de chaque espèce, le renouvellement des populations est essentiellement dû aux meilleurs individus de l'espèce.

De cette théorie d'évolution, l'homme acquit la connaissance sur ses propriétés mathématiques et il définit pour l'informatique, les algorithmes d'optimisation appelés "Algorithmes évolutionnaires". Les algorithmes évolutionnaires sont donc des algorithmes d'optimisations s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle, à savoir les croisements, les mutations, la sélection, etc.

L'algorithme évolutionnaire (AE) est caractérisé par une population de solutions candidates et un processus de reproduction qui permet de combiner les solutions existantes pour produire de nouvelles solutions. Puis la sélection détermine quels individus de la population courante participent dans la nouvelle population. Ce processus est répété plusieurs fois jusqu'à convergence vers des solutions optimales.

On peut distinguer trois grandes classes d'algorithmes évolutionnaires: les algorithmes génétiques, les stratégies d'évolution ou évolutionnistes et la programmation génétique. Ces méthodes se différencient par leur manière de représenter l'information et par leur façon de faire évoluer la population d'une génération à l'autre [35].

II.2. Historique

Le terme d'Algorithme évolutionnaire (AE) est relatif à une classe de méthodes d'optimisation stochastiques qui simulent le processus d'évolution naturelle. Les origines des algorithmes évolutionnaires remontent à la fin des années 50, et depuis 1970, plusieurs méthodes évolutionnaires ont été proposées, principalement concernant les algorithmes génétiques, la programmation génétiques et les stratégies d'évolution [13].

En 1860 Charles Darwin publie son livre intitulé « L'origine des espèces au moyen de la sélection naturelle ou la lutte pour l'existence dans la nature ». Dans ce livre, Darwin rejette l'existence «de systèmes naturels figés», déjà adaptés pour toujours à toutes les conditions extérieures, et expose sa théorie de l'évolution des espèces : sous l'influence des contraintes extérieurs, les êtres vivants se sont graduellement adaptés à leur milieu naturel au travers de processus de reproductions. La sélection naturelle : Sélection des individus les mieux « adaptés » à un milieu donné et qui auront une plus grande faculté de reproduction que les autres. La sélection naturelle soutient donc que les êtres vivants qui s'adaptent le mieux aux conditions naturelles de leur environnement vaincront et survivront. [22]

En 1966 L. J. Fogelintroduisit le concept de la programmation évolutionnaire.

En 1973 I. Rechenberg donna lieu aux stratégies d'évolution.

J.H. Holland, professeur à l'université du Michigan, entreprit avec ses étudiants, en 1975, une vaste étude qui permit de poser les fondements des AG en calquant les principes de Darwin (sélection, croisement, mutation, chromosome, gènes). Il parvint alors, à mettre au point les étapes de l'algorithme et ses principes de codage. Il esquissa aussi les grandes perspectives d'application des AGs. Ces travaux ont suscité un intérêt sans cesse croissant pour les mathématiciens dont Koza qui valida rigoureusement leurs mécanismes. Ensuite, dans les années 90 est apparue la programmation génétique qui introduit notamment des représentations arborescentes [33].

En 1989 David Goldberg publie un ouvrage de vulgarisation des algorithmes génétiques.

Aux années 1990, une programmation d'une panoplie d'algorithmes génétiques transcrit en C ++ , appelée GAlib. Cette librairie contient des outils pour des problèmes d'optimisation en utilisant les AG. Elle est conçue pour servir de support de programmation [37].

II.3. Principe de fonctionnement

Le processus d'optimisation évolutionnaire commence par l'étape de l'initialisation : un nombre fini d'individus p choisis généralement par tirage aléatoire uniforme dans D forment la population initiale P_0.Après évaluation de la population initiale (calcul de la performance), certains individus (les plus performants) sont choisis lors de l'étape de la sélection. L'application des opérateurs de variation (croisement et mutation) permet de créer un nouvel ensemble d'individus, appelé "population d'enfants" (à noter que cette étape est toujours stochastique). Ces enfants vont être évalués à leurs tour et combinés avec leur parents afin de décider lesquels d'entre eux vont remplacer certains parents pour faire partie de la génération suivante, il s'agit de l'étape de remplacement. *(Figure II.1)* illustre le principe général de fonctionnement d'un algorithme évolutionnaire.

A noter que dans la plupart des applications réelles, le coût de calcul des algorithmes évolutionnaires provient essentiellement de l'étape d'évaluation. A titre d'exemple, si l'on souhaite faire évoluer une population de quelques dizaine d'individus pendant quelques dizaine de générations, quelques milliers de calculs de la fonction performance (ou fitnes) doivent être réalisés souvent à travers des évaluations coûteuses [34].

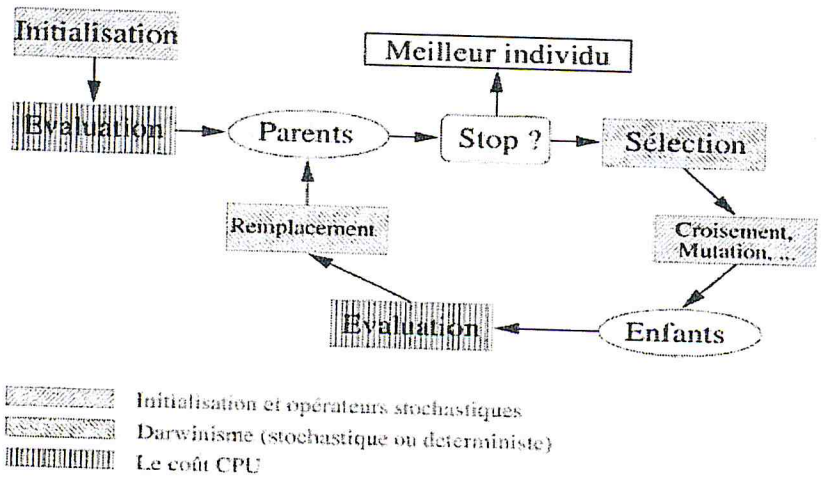


Figure II.1 Principe de fonctionnement d'un AE[34]

II.4. Représentation des individus

On peut distinguer plusieurs types de représentation des individus. Parmi les plus connues, on peut mentionner : la représentation binaire et la représentation réelle.

II.4.1Représentation binaire

Par analogie (imitation) avec la génétique naturelle, les AEs utilisent habituellement des chaînes de bits pour représenter les chromosomes. Ainsi, pour un problème d'optimisation ayant n variables entières, on peut représenter chacune de ces variables par un vecteur binaire de taille égale à k bits (suivant leur domaine) ; on obtiendra ainsi un chromosome final de taille (n * k).

Les premiers résultats de convergence dans la littérature des AEs ont été basés sur de telles chaînes binaires. Ils ont montré que le codage des chromosomes à l'aide de gènes dont l'alphabet possède un faible cardinal était théoriquement plus efficace.

Le codage binaire confère en outre aux AEs une très grande robustesse car il est indépendant du domaine du problème traité et les opérateurs stochastiques standards peuvent être systématiquement utilisés [29].Cependant, ce type de codage présente des inconvénients : malgré son efficacité, on peut constater que pour des problèmes où l'on veut une grande précision dans les résultats, le codage binaire peut facilement devenir inadapté [16]. De plus, deux éléments proches dans l'espace de recherche ne décodent pas nécessairement deux individus voisins en termes de distance de Hamming (nombre de bits différents). Afin d'éviter parfois cet inconvénient les praticiens utilisent un codage de Gray qui conserve une distance de Hamming de " 1 " entre deux individus consécutifs quelconques [15].

II.4.2 Représentation réelle

Le principe de cette représentation consiste à coder directement les variables du problème dans l'individu sans passer par le codage binaire intermédiaire [10] [30]. Ainsi, les individus ne sont plus représentés par des chaînes binaires mais par des vecteurs réels. C'est bien sûr le cas le plus fréquent en calcul numérique ; on parle alors aussi d'optimisation paramétrique.

L'un des avantages majeur de cette représentation est de conserver les variables du problème dans le codage lui-même, ce qui lui permet une meilleure prise en compte de la structure même du problème. Cette représentation directe des paramètres réels nécessite de définir de nouveaux opérateurs de variations génétiques bien spécifiques à ses caractéristiques.

II.5. Les différentes méthodes de sélection

On distingue plusieurs méthodes de sélection, les plus utilisées sont la sélection par roulette proportionnelle et la sélection par tournoi [6].

II.5.1 Sélection par roulette (Roulette Wheel selection) :

Pour chaque individu, la probabilité d'être sélectionné est proportionnelle à son adaptation au problème. Le principe de Roulette Wheel selection est celui de la roue de la fortune biaisée.

Cette roue est une roue de la fortune classique sur laquelle on associe à chaque individu un segment dont la longueur est proportionnelle à sa fitness. On effectue ensuite un tirage aléatoire utilisé dans les roulettes de casinos avec une structure linéaire. Avec ce système, les grands segments, c'est-à-dire les bons individus, seront plus souvent adressés que les petits [6].

II.5.2 Sélection par tournoi

La sélection par tournoi consiste à sélectionner n individus au hasard et à prendre le meilleur parmi ces n individus. On organise autant de tournois qu'il y a d'individus à repêcher. Le nombre n permet de donner plus ou moins de chance aux individus peu adaptés. Avec un nombre élevé de participants, un individu faible sera presque toujours sûr de perdre. Le nombre d'individus par tournoi détermine les paramètres d'exploration (n petit) et d'exploitation (n grand) du bassin génétique [6].

II.5.3 Sélection par rang

La sélection par rang trie d'abord la population par fitness. Ensuite, chaque chromosome se voit associé un rang en fonction de sa position. Le plus mauvais chromosome aura le rang n , le suivant n-1 , et ainsi de suite jusqu'au meilleur chromosome qui aura le rang 1 (pour une population de chromosomes). La sélection par rang d'un chromosome est la même que par roulette, mais les proportions sont en relation avec le rang plutôt qu'avec la valeur de évaluation, c'est à dire les individus choisis sont ceux qui possèdent les meilleurs scores d'adaptation (meilleur rang), le hasard n'entre donc pas dans ce mode de sélection [6].

II.5.4 Sélection « steady-state »

L'idée principale est qu'une grande partie de la population puisse survivre à la prochaine génération. A chaque génération sont sélectionnés quelques chromosomes (parmi ceux qui ont le meilleur coût) pour créer des chromosomes fils. Ensuite les chromosomes les plus mauvais sont retirés et remplacés par les nouveaux. Le reste de la population survie à la nouvelle génération [6].

II.5.5Elitisme

A la création d'une nouvelle population, il y a de grandes chances que les meilleurs chromosomes soient perdus après les opérations d'hybridation et de mutation. Pour éviter cela, on utilise la méthode d'élitisme. Elle consiste à copier un ou plusieurs des meilleurs chromosomes dans la nouvelle génération. Ensuite, on génère le reste de la population selon l'algorithme de reproduction usuel. Cette méthode améliore considérablement les algorithmes génétiques, car elle permet de ne pas perdre les meilleures solutions [6].

II.5.6 Sélection uniforme

La sélection se fait aléatoirement, uniformément et sans intervention de la valeur d'adaptation. Chaque individu a donc une probabilité $1/|P|$ d'être sélectionné, où $|P|$ est le nombre total d'individus dans la population [6].

II.6 Les opérateurs de variation

II.6.1 Le Croisement

En biologie, le terme croisement (Cross-over en anglais) désigne le moment dans l'étape de la fécondation au cours duquel les bagages génétiques du père et de la mère sont échangés. Les chromosomes mâles et femelles s'associent et échangent leurs gènes afin de donner naissance à un individu original mais possédant des caractéristiques provenant des deux parents.

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants *(figure II.2)*.Bien que ce phénomène ait lieu dans toute reproduction, il n'est utilisé dans les EAs qu'avec une probabilité p et ce afin de ne pas faire diverger l'algorithme, ou au contraire le faire converger trop prématurément [35].

Il existe plusieurs types de croisements :

II.6.1.1 Le Croisement à 1-Point

Le croisement à 1-point dont le fonctionnement est le suivant : on suppose que les individus sont codés sur des chaînes de longueur L [35].

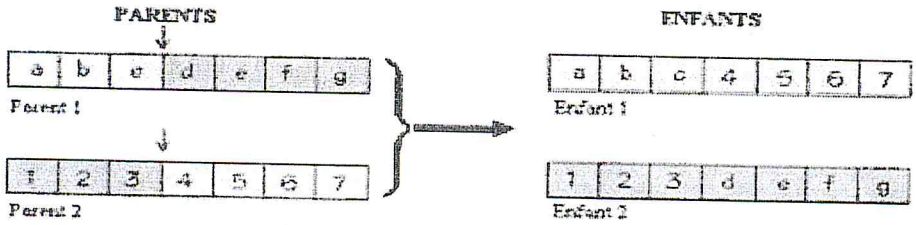


Figure II.2 : Le croisement à 1 point entre deux individus

Le principe est de créer deux nouveaux individus à partir des deux parents, on génère au hasard un entier a et le premier enfant va avoir les a premiers gènes du parent1 et les L– a derniers gènes du parent2, et vice versa pour le deuxième enfant. Dans cet exemple, l'Enfant 1 reçoit les trois premiers gènes du Parent 1 et les quatre derniers gènes du Parent 2 ; l'Enfant 2 quant à lui, hérite les trois premiers gènes du Parent 2 et les quatre derniers gènes du Parent 1. Il existe également le croisement à deux points *(FigureII.3)* et le croisement multipoint [35]

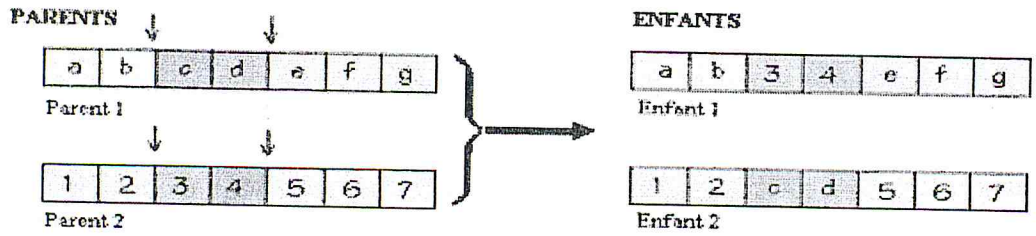
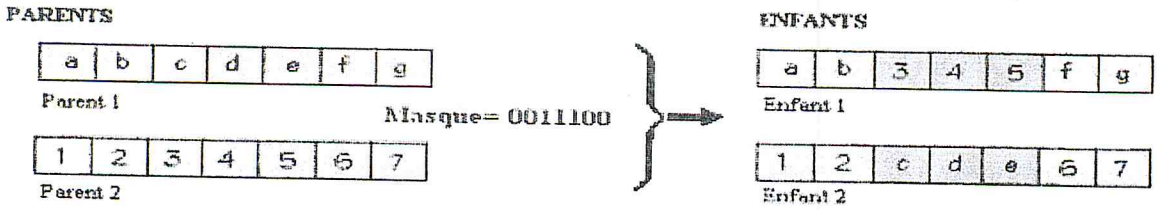


Figure II.3: Croisement à deux points

II.6.1.2Le Croisement Uniforme

Dans ce type de croisement, on utilise un masque de croisement (mask), qui consiste en un vecteur généré aléatoirement, de longueur identique aux chaînes parents, et composé de 0 et 1. Lorsque le bit du masque vaut 0, le premier enfant hérite le bit correspondant du premier parent, sinon il hérite celui du second parent. Le second enfant est le complémentaire du premier. Ce croisement peut être considéré comme une généralisation du croisement multipoint sans connaissance préalable du point de croisement *(Figure II.4)* [35].



FigureII.4 : Croisement uniforme

II.6.1.3 Croissement SBX

Le croisement binaire simulé (SimulatedbinaryCrossover) reproduit les mécanismes du croisement standard à un point utilisé lorsque les variables objets sont représentés sous la forme de chaînes binaires [Deb99a]. A partir de deux parents $P_1(i)$ et $P_2(i)$, ce croissement génère deux enfants parents $C_1(i)$ et $C_2(i)$ par l'intermédiaire de la relation suivante [25].

$$\begin{cases} c_1(i) = 0.5[(1+\beta)p_1(i) + (1-\beta)p_2(i)] \\ c_2(i) = 0.5[(1-\beta)p_1(i) + (1+\beta)p_2(i)] \end{cases} \text{(II.1)}$$

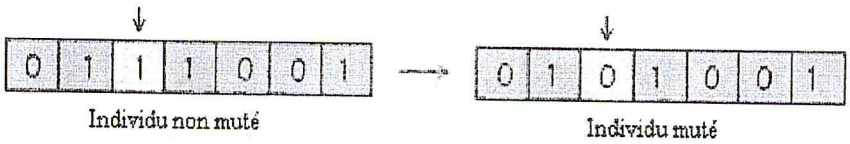
β représente un facteur de dispersion défini par : [25]

$$\beta = \begin{cases} (2u)^{\frac{1}{\eta+1}} & si\ u < 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{\eta+1}} & ailleurs \end{cases} \quad \text{(II.2)}$$

Où u est une variable aléatoire uniformément répartie dans l'intervalle [0,1] et η un paramètre réel non négatif qui caractérise la forme de la distribution des enfants par rapport aux parents de forte valeur η engendre une forte probabilité de retrouver un enfant dans le voisinage local des parents. Tout comme les stratégies d'évolution. Le croisement SBX possède des propriétés intéressantes d'auto-adaptation [25].

II.6.2La Mutation

Cet opérateur génétique consiste à modifier le génotype d'un individu. La mutation a pour but de garantir l'exploration de l'espace de décision. Les propriétés de convergence des MOEAs sont fortement dépendantes de cet opérateur sur le plan théorique, et un algorithme peut converger rien qu'en utilisant des mutations. Pour les problèmes discrets, l'opérateur de mutation consiste généralement à tirer aléatoirement un gène dans le chromosome et à le remplacer par une valeur aléatoire *(Figure II.5)*[35].



FigureII.5: La mutation d'un gène vers un autre

II.6.3 La Mutation polynomiale

Deb et Agrawal [13] ont proposé un opérateur de mutation polynôme avec un paramètre d'index défini par l'utilisateur (η_m) Basé sur une étude théorique, ils ont conclu que η_m induit un effet d'une perturbation de O ((b-a) / η_m) dans une variable ou a et b sont les bornes inférieur et le borne supérieure du variable, Ils ont également constaté qu'une valeur $\eta_m \in$ [20,100]est suffisante dans la plupart des problèmes qu'ils ont essayé

Dans cet opérateur, une distribution de probabilité polynômiale est utilisée pour perturber une solution dans le voisinage d'un parent. La distribution de probabilité à la fois à gauche et à droite d'une valeur de la variable est ajustée de telle sorte que aucune valeur en dehors la gamme spécifiée [a, b] Est créé par l'opérateur de mutation.

Une solution père proposée p ∈ [a, b], p' solution muté pour une variable particulière est criée pour un nombre aléatoire u crée au sien [0 ,1] comme suit [25] :

$$p' = \begin{cases} p + \overline{\delta_L}\left(p - x_i^{(L)}\right), & si\ u \leq 0.5, \\ p + \overline{\delta_R}\left(x_i^{(u)} - p\right), & si\ u > 0.5. \end{cases} \qquad \text{(II.3)}$$

Alors, l'un des deux paramètres $\bar{\delta}_L ou \bar{\delta}_R$ est calculé comme suit [25] :

$$\overline{\delta_L} = (2u)^{1/(1+\eta_m)} - 1 \qquad si\ u \leq 0.5; \qquad \text{(II.4)}$$

$$\overline{\delta_R} = 1 - (2(1-u))^{1/(1+\eta_m)} \qquad si\ u > 0.5. \qquad \text{(II.5)}$$

II.7. Types d'Algorithmes évolutionnaires

On distingue quatre grandes familles historiques d'algorithme *(Figure II.6)* et les différences entre elles ont laissé des traces dans le paysage évolutionnaire actuel, en dépit d'une unification de nombreux concepts [37].

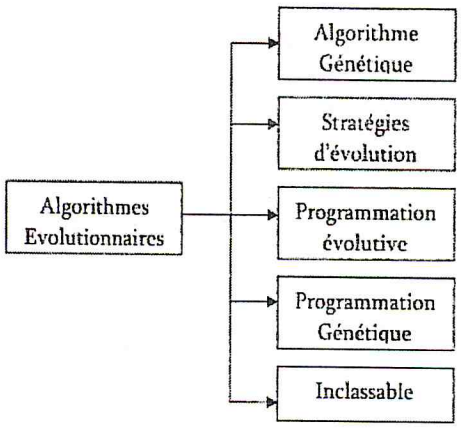


Figure II.6 : Les différentes branches des algorithmes évolutionnaires

II.7.1. Les algorithmes génétiques

II.7.1.1L'optimisation multi-objectif et les algorithmes génétiques

A partir duschéma fonctionnel desalgorithmes génétiques multi objectifs(**voire la Figure II.7**), on définit quatre composantes à spécifier pour adapter ces algorithmes à la résolution d'un problème donné :

1) une représentation génétique des solutions du problème,
2) une façon de créer une population initiale de solutions, une fonction d'évaluation jouant le rôle de l'environnement, distinguant les solutions en termes de leur adaptation,
3) des opérateurs génétiques qui modifient la composition des enfants pendant la reproduction,
4) des valeurs pour les divers paramètres que l'algorithme génétique utilise (taille de la population, probabilités d'application des opérateurs génétiques, etc). [5]

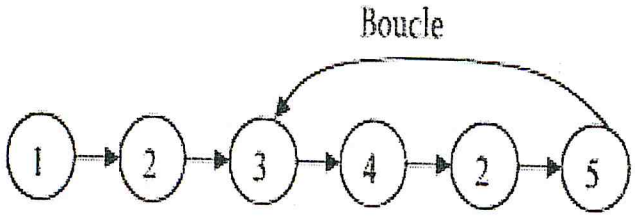


Figure II.7 : Le fonctionnement d'un algorithme génétique multi-objectif [40]
(1)Initialisation de la population, (2) Evaluation de l'efficacité des individus de la population. (3)Croisement, (4) Mutation, (5) Sélection.

Nous allons maintenant nous intéresser aux diverses méthodes évolutionnaires qui ont été mises en œuvre pour le traitement de problèmes d'optimisation multi-objectifs. Ces méthodes se décomposeront en deux familles :

- ✓ Une première famille comportera les algorithmes dits « non agrégatifs ».
- ✓ La deuxième famille comportera les algorithmes « agrégatifs » [37].

II.7.1.2 Les méthodes non agrégatives

La méthode VectorEvaluatedGeneticAlgorithm (VEGA)

Cette méthode permet de traiter un problème d'optimisation multi-objectifs sans avoir à agréger les fonctions objectives en une seule fonction.

La *(Figure II.8)*représente les différentes séquences de fonctionnement de la méthode. Sur cette figure, on a représenté les différents types de populations que l'on traite au cours du déroulement de la méthode VEGA (soit un ensemble d'individus, soient des groupes d'individus).

Etape 1 : Itération i: Initialisation d'une population de taille N.

Etape 2 : Création de k groupes (sous population).

Etape 3 : Calcul des efficacités.

Mélange des individus.

Etape 4: On applique l'algorithme génétique classique (croisement, mutation et sélection). Puis on passe à l'itération suivante (i + 1) [37].

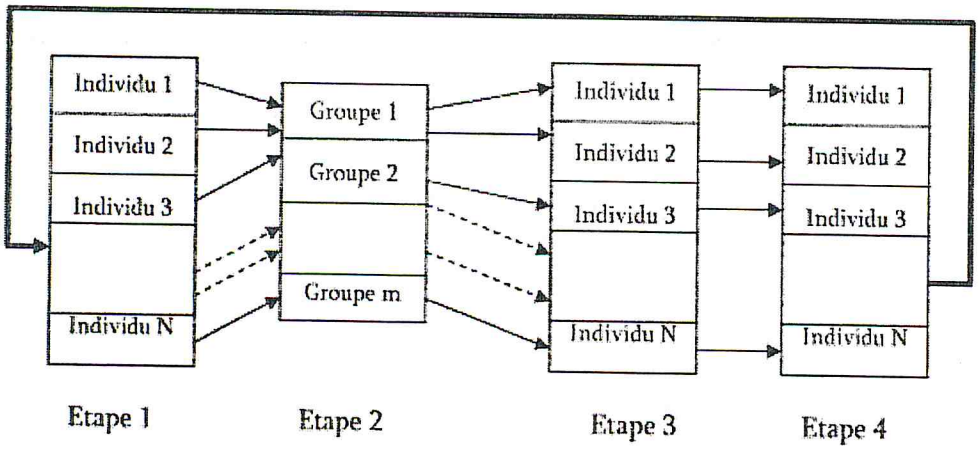


Figure II.8: Principe de l'algorithme VEGA [40]

II.7.1.3Les méthodes agrégatives

La méthode Multiple Objective GeneticAlgorithm (MOGA)

Cette méthode populaire consiste à ramener un problème multi-objectif à unproblème d'optimisation d'une combinaison linéaire des objectifs initiaux. Cettecombinaison est caractérisée souvent par une pondération qui traduit l'importancerelative des objectifs.

$$F_m(x) = \sum_{m=1}^{M} w_m f_m(x) \qquad \text{(II. 6)}$$

Où les $w_m \leq 0$ sont les points respectifs des M objectifs à optimiser.

Cependant, cette méthode permet de résoudre uniquement des problèmes simples avec des fronts de Pareto convexe. Pour la plus part des problèmes non-linéaires ou des problèmes dont la surface de Pareto contient des régions non-convexes cette méthode ne permet pas de trouver les solutions de Pareto [34].

II.7.2 La programmation évolutive

La programmation évolutive a été initialement introduite pour simuler l'intelligence qui est définie sur l'hypothèse suivante : la caractéristique principale de l'intelligence est la capacité d'adaptation comportementale d'un organisme à son environnement. Aujourd'hui la programmation évolutive s'est adaptée à l'optimisation combinatoire.

La programmation évolutive s'appuie sur un codage approprié du problème à résoudre et sur les opérations de mutation adaptées au codage. Le codage d'un tel algorithme dépend du problème à résoudre. Par exemple, pour un problème d'optimisation dans le domaine des réels, les individus d'une population seraient des vecteurs de réels.

Un cycle d'évolution typique pour la programmation évolutive est le suivant : chaque configuration de la population courante est copiée dans une nouvelle population. Les configurations sont ensuite mutées, conduisant à de nouvelles configurations.

L'ensemble des configurations entre ensuite dans une étape de compétition pour survivre dans la génération suivante [39].

II.7.3 Les stratégies d'évolution

Les stratégies d'évolution sont conçues dès le départ pour résoudre des problèmes d'optimisation continus. Dans ces algorithmes, les individus sont représentés par des points (vecteurs de réels), et les seuls mécanismes utilisés sont la mutation et la sélection [39]; elles sont des algorithmes itératifs dans lesquels un parent génère un enfant *(1+1)*-ES. Le meilleur des deux survit et devient le parent de la génération suivante. La génération de ce processus a donné les algorithmes *$(\mu+\lambda)$-ES* dans les μ parents génèrent λ enfant. Les μ meilleurs survivent [32].

II.8.Techniques Avancées d'Amélioration des MOEAs

Les algorithmes évolutionnaires multi objectifs (MOEAs) ont montré une certaine efficacité dans la résolution des problèmes de la MOO. Mais vu leur aspect stochastique, les MOEAs peuvent ne pas conserver les bonnes solutions rencontrées pendant leur exécution et les rendre disponibles à la fin de l'algorithme, de ce fait est née la notion d'élitisme qui préconisent que les meilleurs individus de la population actuelle seront pris dans la prochaine génération. Une autre difficulté que peut rencontrer un MOEA est qu'un individu ayant une très bonne valeur de fitness, a tendance à se multiplier aux dépens des autres individus de la population et par conséquent le MOEA va se coincer autour d'un seul optimum et fournir à la fin d'exécution un font Pareto mal distribué. D'un autre côté, il y a une récente tendance à faire combiner les MOEAs avec d'autres méthodes de recherche et d'exploration de l'espace des solutions en vue d'obtention de résultats meilleurs que ceux obtenus par les MOEAs seuls. Ces méthodes hybrides ont montré leur efficacité pour trouver des solutions approchées satisfaisantes pour un grand nombre de problèmes. En général les améliorations suggérées et apportées aux MOEAs évoluent autour de trois axes : l'élitisme, le maintien de la diversité et l'hybridation (Coopération) [35].

II.8.1. L'Élitisme

Une stratégie élitiste consiste à conserver au moins un individu de la population de la génération précédente, dans la génération suivante. [3]

II.8.1.1 Les méthodes élitistes

II.8.1.1.1 Pareto ArchivedEvolutionaryStrategy (PAES)

En 1999, Knowles et Corne propose la méthode PAES (Pareto Archived Evolution Strategy). Cette méthode a été développée initialement comme uneméthode de recherche locale dans un problème de routage d'information off-line [2].

PAES est un algorithme classé stratégie d'évolution. Les auteurs [Knowles et Corne, 00a] de cet algorithme ont été motivés pour l'utilisation de la stratégie d'évolution car ils ont remarqué que lors de la résolution des problèmes de MOO dans le domaine de télécommunication, les stratégies basées sur le voisinage telles que la recherche taboue et le recuit simulé, donnent de meilleurs résultats que les approches qui utilisent une population de solutions.

Cet algorithme commence par générer aléatoirement une solution initiale et puis, une solution candidate est produite dans chaque itération au moyen de mutation.

Une archive externe (de taille limitée) est maintenue pour rassembler les solutions non-dominées. La solution candidate est écartée si elle est dominée par la solution courante ou n'importe quelle autre solution dans l'archive externe. La solution candidate est ajoutée à l'archive et devient la solution courante si elle domine la solution actuelle. Si aucune d'elles ne domine l'autre, la décision sur la solution qui va devenir la solution courante et si elle va être ajoutée ou pas à l'archive est basée sur le mécanisme d'encombrement dit « crowding » D'autres variantes de cet algorithme avec la population ont été également proposées [23].

```
Génération aléatoire d'une solution S
Ajouter S à l'archive ;
Répéter
Production d'une solution C par mutation de S
Evaluation de C
Si  S domine C  alors
        Ecarter m ;
Sinon si C domine S
        Remplacer S par C ajouter m à l'archive ;
    Sinon si C est dominé par un membre de l'archive alors
            On écarte C
        Sinon  S= meilleur (S,C,archive)
Finsi
Finsi
Finsi
Jusqu'à condition d'arrêt valide
Fin
```

La fonction Meilleur() aura pour rôle de déterminer tout d'abord si l'archive n'est pas pleine dans ce cas ajoute C à l'archive. Dans le cas contraire, elle détermine la solution à supprimer de l'archive pour pouvoir insérer C. La solution à supprimer doit appartenir à une région plus encombrée que celle de C. Puis elle détermine laquelle de deux solutions S et C qui va devenir la solution courante. Bien sûr celle qui appartient à une zone moins encombrée est sélectionnée [35].

II.8.1.1.2 Pareto Envelope based Selection Algorithm (PESA)

La méthode PESA (Pareto EnvelopebasedSelectionAlgorithm) a été également proposée par Knowles et Corne [39]. Alors que PAES est basée sur une stratégie d'évolution, PESA est une méthode basée sur les algorithmes génétiques. Elle reprend approximativement le principe de crowding développé dans PAES [2].

Elle définit deux paramètres concernant la taille des populations d'individus : P_1(taille de la population interne) et P_E(taille de la population externe ou archive).

Une solution courante de $P_I$peut entrer dans l'archive P_E si elle est non dominée dans P_I et si elle est non dominée dans $P_E$une fois la solution insérée dans l'archive, on supprime tous les membres de l'archive qu'elle domine .Si l'ajout crée un dépassement de capacité deP_E, alors Le membre de l'archive ayant le paramètre '*squeeze_factor*' le plus élevé est supprimé [32].

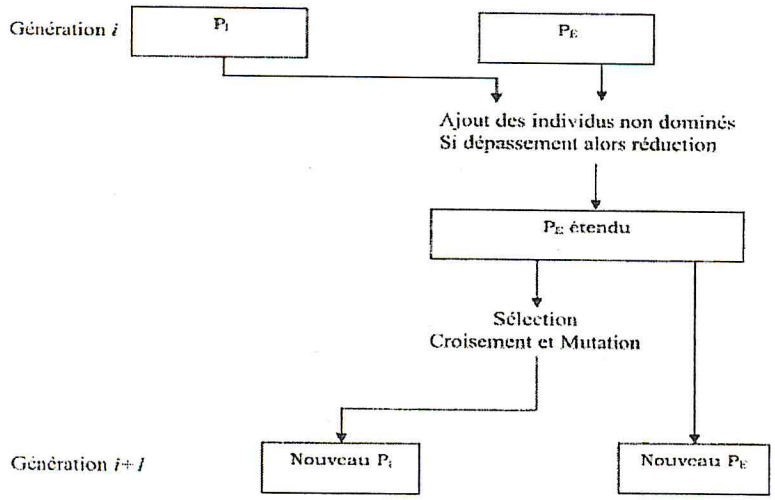


Figure II.9 : Schéma de fonctionnement de PESA [32]

II.8.1.1.3 Non Dominated Sorting Genetic Algorithm II (NSGAII)

Ont proposé une nouvelle version de l'algorithme NSGA est le NSGA-II. Il est considéré plus efficace que son prédécesseur.

NSGA-II est un algorithme élitiste n'utilisant pas d'archive externe pour stocker l'élite. Pour gérer l'élitisme, NSGA-II assure qu'à chaque nouvelle génération, les meilleurs individus rencontrés soient conservés pour la génération suivante [35].

L'algorithme NSGAII est un algorithme évolutif multi-objectif, établissant les rapports de dominance entre les individus et offrant une méthode de tri particulièrement rapide des chromosomes. Cet algorithme utilise la mesure du surpeuplement autour des individus pour assurer la diversité dans la population. Le principe de cet algorithme est illustré par la *Figure II.10.*

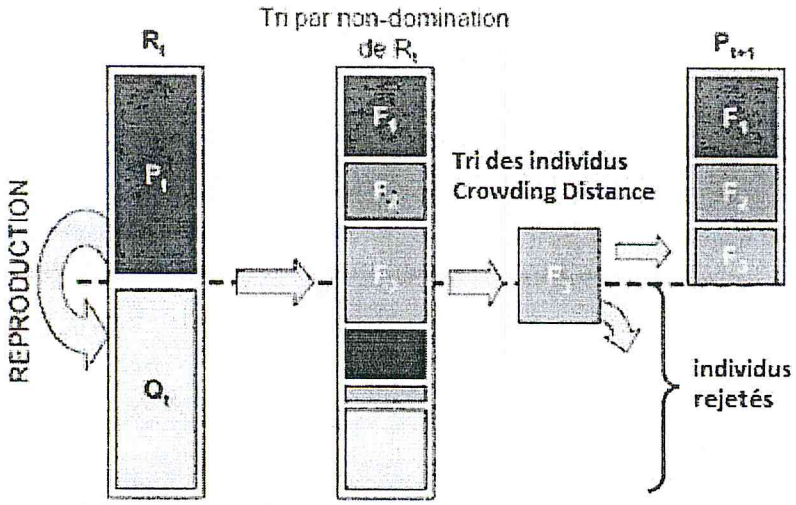


Figure II.10 : Principe de fonctionnement de l'algorithme NSGAII [27]

Au début, une population initiale est générée aléatoirement, puis elle subit un tri en utilisant le concept de la non-domination. Chaque solution se voit affecter une force, ou rang, égal à son niveau de non-dominance (1 pour le meilleur niveau, 2 pour le niveau suivant,…,etc.). L'étape de reproduction consiste en un tournoi pour la sélection des parents. Deux individus de la population sont choisis aléatoirement dans la population, le tournoi est basé sur une comparaison de la domination sous contraintes des deux individus.

Pour une génération t donnée, on crée $R_t = P_t \, UQ_t$, où Q_t étant la population enfants de la population précédente R_t (générés à partir des parents à travers les opérateurs de croisement et de mutation), R_t inclut les individus de P_t , ce qui assure le caractère

élitiste de l'algorithme NSGAII. La population $R_t$contient 2N individus (elle est composée de N parents et N enfants). $R_t$subit ensuite un tri en utilisant le concept du non dominance de Pareto.

Les individus sont regroupés dans des fronts de non-dominance tels que F 1représente les individus de rang 1, F2 les individus de rang 2, … etc. L'objectif suivant est de réduire le nombre d'individus de 2N dans la population R_t pour obtenir une population P_{t+1} de taille N. Si la taille de F1 est inférieure à N, alors tous les individus de F1 sont conservés. Il en est de même pour les autres fronts tant que le nombre d'individus conservés ne dépasse pas la taille N. Si l'on prend l'exemple de la figure II.7, les fronts F1 et F2 sont intégralement conservés mais la conservation du front F3va entraîner un dépassement de la taille N de la population P_{t+1} . Il faut alors procéder à une sélection des individus de F3 à conserver.

Dans ce cas l'algorithme NSGAII fait intervenir un mécanisme de préservation de ladiversité de la population basé sur l'évaluation de la densité des individus autour de chaque solution, à travers une procédure de calcul de «Distance de surpeuplement». Une faible valeur de la Distance de proximité pour un individu correspond à un individu «bien entouré». On procède alors à un tri décroissant selon cette distance de proximité pour retenir des individus du front F 3 et éliminer ainsi les individus des zones les plus denses. On complète de cette façon la population P_{t+1} . Les individus ayant des valeurs extrêmes pour les critères sont également préservés par ce mécanisme, permettant ainsi de conserver les bornes extérieures du front de Pareto.

À la fin de cette phase, la population P_{t+1} est créée. Puis une nouvelle populationQ_{t+1} est générée par reproduction à partir de R_{t+1} . On poursuit itérativement la procédure décrite ci-dessus jusqu'à la satisfaction de critère d'arrêt fixé par l'utilisateur.

De point de vue général, le NSGAII permet de maintenir l'élitisme et la diversité sans ajouter de paramètres supplémentaires, tout en utilisant un algorithme séduisant par sa simplicité avec un minimum de paramètres de réglage [27].

Calcul des distances de surpeuplement (crowding distance)

Pour estimer la densité des solutions voisines d'une solution i dans un ensemble non-dominé F la quantité d_i appelés ici «Distance de surpeuplement». Est calculée de façon suivante.

Calcul des distances de surpeuplement dans un ensemble non-dominé F

1. Soit l=| F | . soit d'abord d_i=0 pour toute solution i de F . soit le compteur d'objectifs m=1

2. Pour l'objectif m , réordonner l'ensemble F de façon que les valeur de f_m sur ses éléments diminues. Soit $I^m = sort_{[f_{m>}]}$ (F) le vecteur des indices, c'est-à-dire I_i^m dénote l'indice de la solution i dans la liste ordonnée selon l'objectif m .

3. Pour chaque solution i tel que :$2 \leq I_i^m \leq (l-1)$ mettre à jour la valeur de d_i comme suit :

$$di \leftarrow di + \frac{f_m^{I_i^{m+1}} - f_m^{I_i^{m-1}}}{f_m^{max} - f_m^{min}} \qquad (II.7)$$

et associer les valeurs de distance très grandes aux solutions sur les extrémités de F c'est-à-dire si $I_i^m = 1$ *ou* $I_i^m = l$, di = inf

4. Si m=M , la procédure est terminé , sinon incrémenter le compteur d'objectifs $m \leftarrow (m+1)$ et retourner à l'étape 2 [26]

II.8.1.1.4 Strength Pareto Evolutionary Algorithm II (SPEA-II)

Cet algorithme est proposé par [Zitzler et Thiele, 99]. C'est un algorithme qui implante trois techniques communes à d'autres approches telles que : l'utilisation de la dominance au sens de Pareto pour évaluer et sélectionner les individus, l'utilisation d'une population secondaire (archive externe) pour stocker les solutions non-dominées et l'utilisation d'une technique dite« clustering » pour maintenir la diversité basée sur la notion de niche. *(Figure II.11)* montre le fonctionnement de SPEAII [35]:

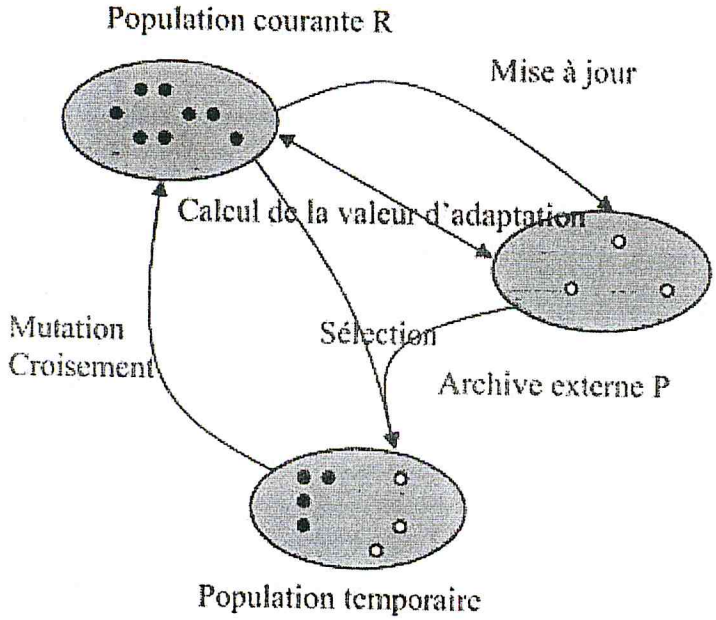


Figure II.11 : Illustration du déroulement [37]

Pour commencer, SPEAII crée aléatoirement une population initiale P_0 à partir de laquelle il rempli l'archive externe P par les individus non dominés dans P_0. À chaque itération, il calcul la valeur fitness des individus des deux populations. Selon les valeurs de fitness obtenues, les individus vont être sélectionnés pour les opérations de croisement et mutation afin de générer de nouveaux individus. Avant d'entamer une nouvelle génération, l'archive est mis à jour par les individus non dominés nouvellement produits. Si après cette mise à jour, des individus se révèlent dominés, ils seront automatiquement supprimés de l'archive.

Si la taille de l'archive après insertion, excède la taille permise, une réduction par« clustering » est alors effectuée [35].

➢ Algorithmes

Voilà dans l'organigramme suivant, le schéma de la méthode de SPEAII

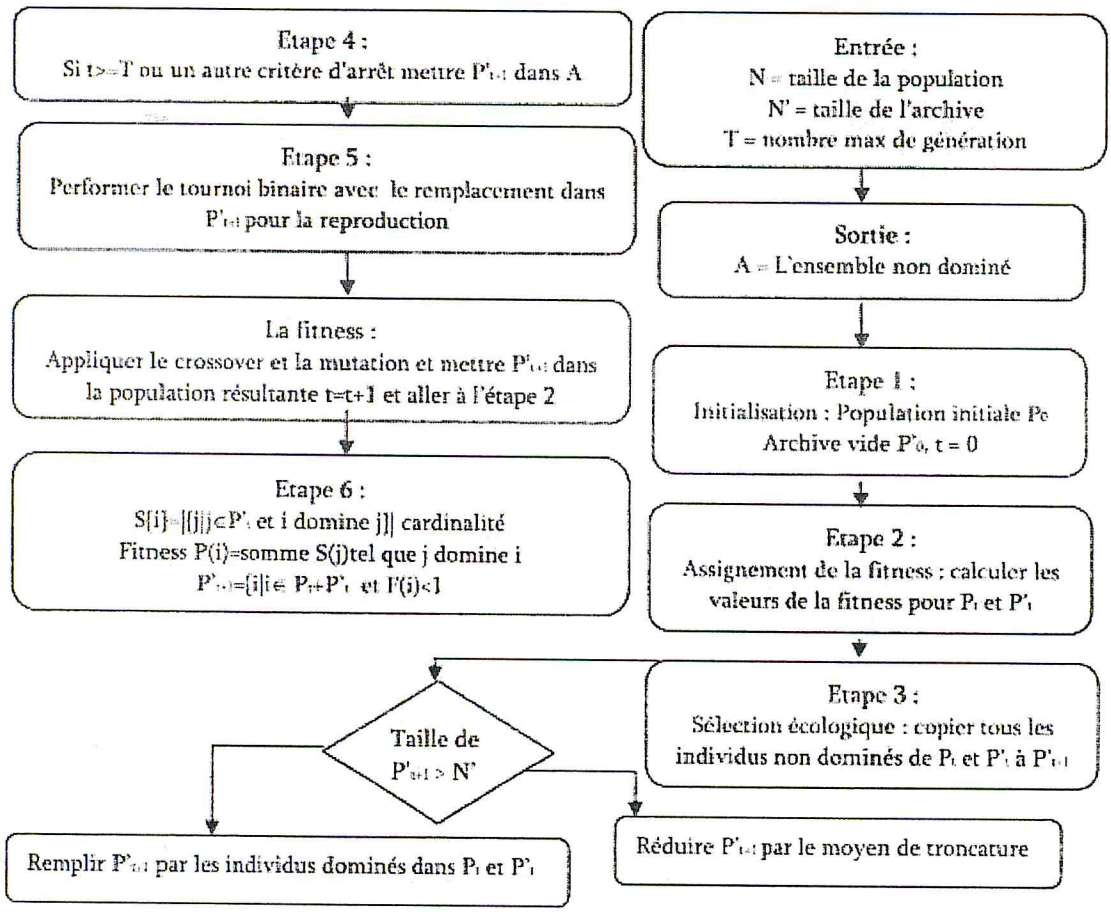


Figure II.12 : Le Principe de l'algorithme SPEA II [37]

Algorithme de clustering

1. Toute solution i appartient à son cluster C_i={i} $Soit\ C = \{C_1, \ldots\ldots\ldots, C_{N'trop}\}$l'ensemble de tous les clusters
2. Si $|C| \leq N'$, passer à l'étape 5, sinon , passer à l'étape 3
3. Pour tout pair de cluster, calculer les distances utilisant l'équation ci-dessus. Trouver le pair (i_1, i_2) qui correspond à la distance minimale
4. Fusionner les clusterC_{i_1} et C_{i_2}. retourner à l'étape 2

5. Dans chaque cluster, trouver la solution dont la distance moyenne des autres solutions du même cluster est minimale et éliminer toutes les autres solutions du cluster [26]

II.8.2 Le Maintien de la Diversité

La diversité est une notion déjà importante lors de l'optimisation de problèmes à un objectif, elle devient prépondérante lorsque l'on traite de problèmes multi-objectifs.

Maintenir un certain degré de diversité dans la population d'un algorithme évolutionnaire consiste à éviter que la population ne converge prématurément vers une petite zone de l'espace de recherche ou de l'espace des objectifs. En effet, s'il n'existe pas de mécanisme de contrôle de la diversité, les opérations de sélection vont privilégier trop vite certains individus meilleurs à cette étape de la recherche. Cette convergence prématurée a comme effet de limiter la recherche à un sous-ensemble plus restreint de l'espace de recherche, qui peut ne contenir aucune solution optimale. Dans le cas de problèmes multi-objectifs, converger prématurément rend impossible la découverte de l'intégralité des frontières de Pareto.

En effet, les individus de la population se focaliseront sur une partie des frontières de Pareto, et ne se répartiront pas sur la totalité de ces frontières. Pour remédier à ce problème, de nombreuses techniques ont été développées. Celles-ci nuent sur la pression de sélection, afin de privilégier certains individus, permettant d'obtenir une population plus diversifiée. Cependant, ces techniques ajoutent un coût calculatoire non négligeable pour l'algorithme, elles doivent donc être choisies avec soin [37].

Quelques techniques utilisées pour maintenir la diversité sont :

- **La Fonction de Partage ou Sharing** : Le sharing consiste à modifier la valeur de coût d'un individu (calculée uniquement à partir de la fonction objective du problème). C'est cette nouvelle valeur qui sera utilisée comme valeur d'adaptation par l'opérateur de sélection [12].
- **Le crowding :** Holland a été le premier à suggérer l'utilisation de l'opérateur de "crowding" dans la phase de remplacement des AGs, pour identifier les situations dans lesquelles de plus en plus d'individus dominent les niches écologiques [19].

- **La réinitialisation :** La réinitialisation est une technique largement utilisée par toutes les métaheuristiques, les algorithmes évolutionnaires n'échappant pas à la règle. Lors d'approches par population, elle consiste à réinitialiser un certain nombre d'individus de la population, par exemple de manière aléatoire [41].

II.8.3 L'hybridation

Une autre façon d'améliorer les performances d'un algorithme ou de combler certaines de ses lacunes consiste à le combiner avec une autre heuristique .L'idée dans ce cas est de lancer le MOEA en premier lieu afin d'approcher la frontière Pareto, après quoi une autre technique s'occupera du raffinement des solutions trouvées par le MOEA afin de mieux approcher l'ensemble des solutions optimales. Dans ce contexte le MOEA est utilisé pour son pouvoir d'exploration et sa capacité de fournir une approximation de l'ensemble du front Pareto. La technique associée utilise les solutions trouvées par le MOEA comme point de départ, en vue de les améliorer [35].

II.9. Conclusion

Nous avons présenté dans ce chapitre, les concepts de base des algorithmes évolutionnaires multi-objectif. Les MOEAs sont basés sur la théorie de l'évolution de Darwin. Par analogie avec le monde biologique, un MOEA fait évoluer une population d'individus à l'aide de divers opérateurs : sélection, croisements, mutations. La seconde génération a montré ses performances et les MOEAs développés récemment sont tous élitistes. NSGA-II et SPEA-II sont actuellement les MOEAs les plus populaires. Plusieurs applications ont été développées mettant en œuvre les politiques de ces deux algorithmes.

Chapitre III

L'algorithme LMOGA et problème test

III.1. Introduction

Afin d'atteindre notre objectif qui est l'étude de l'effet de l'incorporation de connaissances sur le vrai front de Pareto lors du processus d'initialisation des algorithmes évolutionnaires, nous avons besoin de définir trois points nécessaire pour notre étude :

- La connaissance à incorporer [7]: on prend une solution parmi l'ensemble Pareto optimale du problème considéré. Cette solution pourra être trouvée par n'importe quelle autre méthode comme par exemple : les méthodes classiques de résolution, les métaheuristiques ou autre. Dans notre cas, il s'agit de solutions obtenues selon l'optimalité Lexico_Max_Ordering. Le nombre de solutions de ce type est fini sur l'espace des objectifs, (espace de $\vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \dots \dots f_M(\vec{x}))^T$), et ce nombre correspond au plus à$M!$. Ce dernier $(M!)$est souvent réduit à l'unité et plus particulièrement si l'espace d'image est convexe ou si le front de Pareto correspondant est continu. Ces solutions Lexico_Max_Orderingseront trouvées garce à un algorithme proposé dans [7] dit LMOGA (Lexico-Max-Ordering Gentic Algorithm).
- Les fonctions test [1] [18] [24] [38]: Pour mener des expérimentations sur nos algorithmes, nous avons besoin d'un ensemble de fonctions tests. Cet ensemble doit être soigneusement choisi de façon à mettre à l'épreuve l'efficacité des méthodes étudiées dans diverses situations difficiles. En effet, un "bon" test doit être Tel que:

1. il représente un danger particulier pour la convergence ou pour la diversité;
2. la forme et la position de la surface de Pareto soient connues et les valeurs des Variables de décisions correspondantes soient faciles à trouver [28].

   Dans cette étude nous avons choisi sept tests bi-objectifs, il s'agit de cinq fonctions ZDT (cette appellation est relative aux auteurs qui les ont proposées : E.Zitzler, K. Deb, L.Thiele) et deux autres trouvées la littérature. Notre choix s'est fixé sur ces tests car ils ont servi comme une base commune pour la comparaison des Algorithmes Evolutionnaires Multi-Objectif existants et pour l'évaluation des nouvelles techniques.

- Les mesures de performances : Il s'agit de métriques utilisées pour mesurer la qualité des résultats obtenues par un algorithme évolutionnaire multi objectif (AEMO). Ces métriques peuvent être divisées en celles destinées à la qualité de convergence vers le

front de Pareto et celles destinées à la distribution de la diversité au niveau de ce front.

III.2. Algorithme LMOGA (Lexico-Max-Ordering Gentic Algorithm)

III.2.1 Présentation de la méthode

Cet Algorithme (LMOGA) [7] nous permis d'avoir les solutions LMO (au sens de l'optimalité LEXICO-MAX-ORDEING)sans passer par la recherche de tout le Front comme dans le cas de NSGAII ou SPEAII [7]. Selon les expérimentations de cet algorithme dans [7], les fonctions testes ZDT que nous allons présenter ci-dessous admettent des solutions lexico-max-ordering uniques.

Ci-dessous nous présentons l'algorithme LMOGA (Lexicographique Max Ordering Genetic Algorithm).

- ✓ Initialisé une population initiale p(0) faisable et initialiser un compteur d'itération t=0
- ✓ Evaluation de fonctions objectives
- ✓ Trie des éléments en se basant sur la dominance Lexicographique Max Ordering
- ✓ Assignation d'une efficacité

For i=1 to G

- ✓ Sélection aléatoire proportionnelle à l'efficacité
- ✓ Croissement
- ✓ Mutation
- ✓ Evaluation des fonctions objectives
- ✓ Trie des éléments en se basant sur la dominance Lexicographique Max Ordering
- ✓ Assignation d'une efficacité

End For

III.3. Les problèmes tests

Pour valider un algorithme, nous avons besoin d'un ensemble de fonctions tests [1], Plusieurs fonctions tests ont été utilisées pour valider les performances du modèle proposé. Ces fonctions ont plusieurs caractéristiques qui les rendent idéales pour tester la capacité de l'approche proposée à identifier la frontière optimale de Pareto. Il faut noter que ce sont les fonctions de benchmark les plus utilisées dans la littérature.

III.3.1 ZDT1

Le premier de cet ensemble de tests est le plus simple, le front de Pareto correspondant étant continu, convexe et avec la distribution uniforme des solutions le long du front. [1]

$$f_1(x) = x_1 \qquad (III.1)$$

$$f_2(x) = g(x)\left(1 - \sqrt{x_1/g(x)}\right) \qquad (III.2)$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)\Big/(n-1) \qquad (III.3)$$

Ou $x_i \in [0, 1]$ pour tout i = 1, . . ., n et n = 30. Le front de Pareto de ce problème est présenté dans la figure III.1

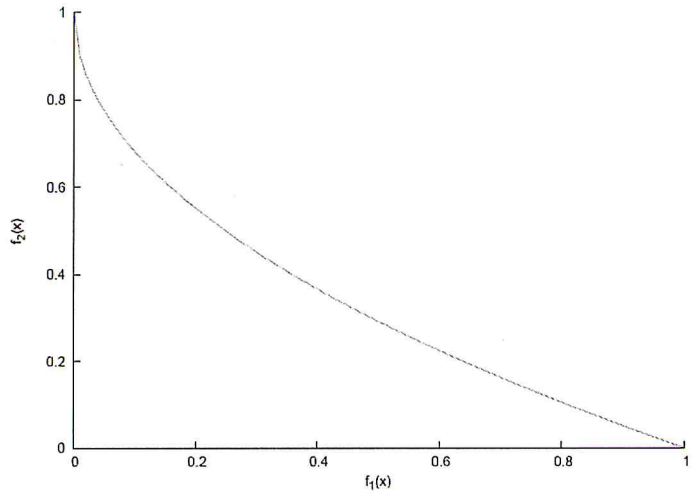


Figure III.1 : La solution de ZDT1 [1]

III.3.2 ZDT2

La difficulté de ce problème se présente dans la non-convexité du front de Pareto [38].

$$f_1(x) = x_1 \qquad (III.4)$$

$$f_2(x) = g(x)(1\ (x_1/g(x))^2) \qquad (III.5)$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)\Big/(n-1) \qquad (III.6)$$

Ou $x_i \in [0, 1]$ pour tout i = 1, . . ., net n = 30. Le front de Pareto de ce problème est présenté par *(Figure III.2)*.

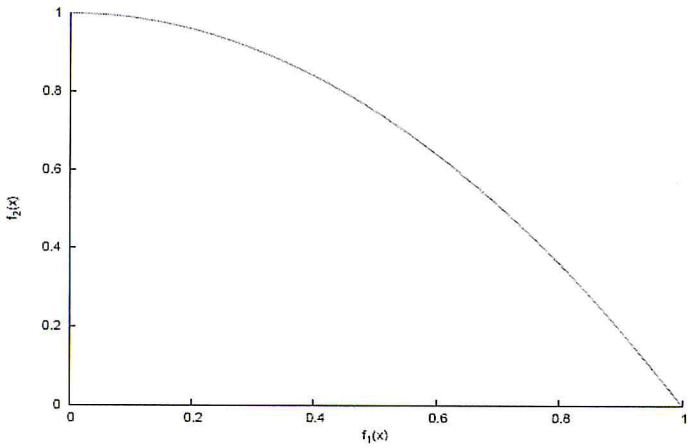


Figure III.2 : La solution de ZDT2 [1]

III.3.3 ZDT3

La difficulté de ce problème c'est que le front de Pareto est discontinu [1].

$$f_1(x) = x_1 \qquad (III.7)$$

$$f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right] \qquad (III.8)$$

$$g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)\Big/(n-1) \qquad (III.9)$$

Ou $x_i \in [0, 1]$ pour tout i = 1, . . ., n et n = 30. Le front de Pareto de ce problème est présenté par *(Figure III.3)*

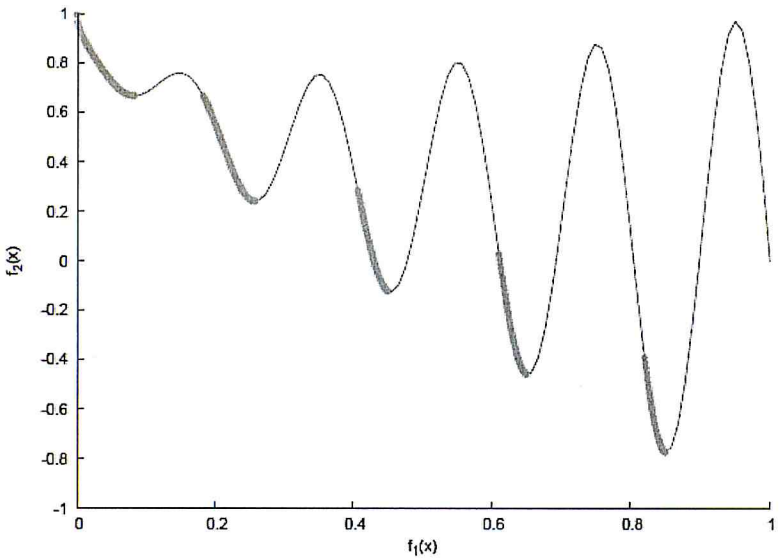


Figure III.3 : La solution de ZDT3 [1]

III.3.4 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 multimodale [38].

$$f_1(x) = x_1 \qquad (III.10)$$

$$f_2(x) = g(x)\left[1 - \sqrt{x_1 g(x)}\right] \qquad (III.11)$$

$$g(x) = 1 + 10(n-1) + \sum_{i=1}^{n}[x_i^2 - 10\cos(4\pi x_i)] \qquad (III.12)$$

Ou $x_1 \in [0, 1]$, $x_i \in [-5, 5]$ pour tout i = 2, . . ., n et n = 10. Ce test comporte (21^9-1) fronts locales. Quelques-uns de ces derniers ainsi que le front global sont présentés dans la *(Figure III.4)* [1].

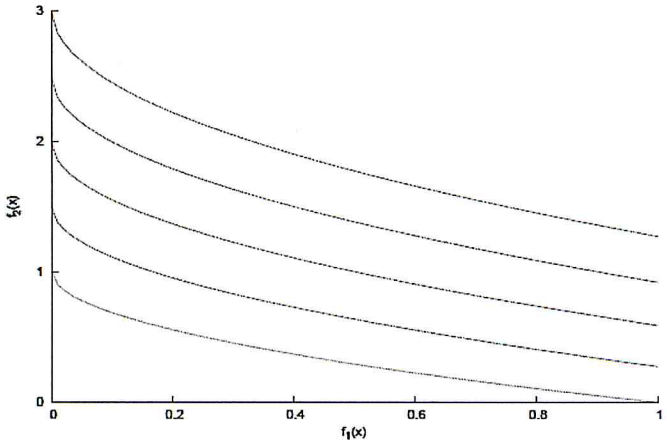


Figure III.4 : La solution de ZDT4 [1]

III.3.5 ZDT6

La particularité de ce problème c'est que les solutions optimales ne sont pas Uniformément distribuées le long du front de Pareto. Cet effet est dû à la non-linéarité de la fonction f1. [1] [38]

$$f_1(x) = 1 - \exp(-4\,x_1)\,sin^6(4\,\pi x_1) \qquad (III.13)$$

$$f_2(x) = g(x)[1 - (f_1(x)/g(x))^2] \qquad (III.14)$$

$$g(x) = 1 + 9\left[\left(\sum_{i=2}^{n} x_i\right)/(n-1)\right]^{0.25} \qquad (III.15)$$

Ou $x_i \in [0, 1]$ pour tout i = 1, . . ., n et n = 10. Ce test comporte (21^9-1) fronts locales.
Le front de Pareto de ce problème est présenté dans la *(Figure III.5)* ainsi que les solutions correspondantes à 100 valeurs de x1 uniformément distribuées sur le segment [0, 1].

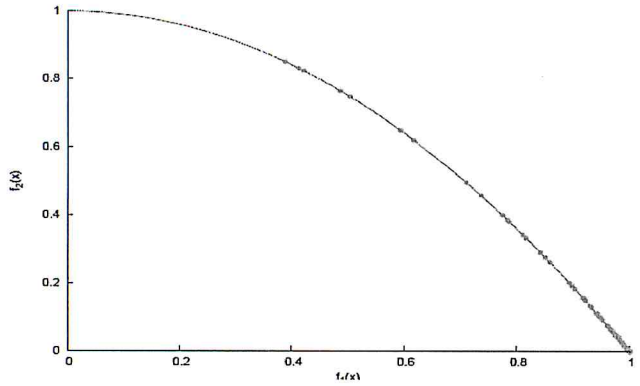


Figure III.5 : La solution de ZDT6 [1]

III.3.6 SCH :

Le front de Pareto de ce problème teste est un front convexe voilà la formule ci-dessous [24]

$$f_1(x) = x^2 \qquad (III.16)$$

$$f_1(x) = (x-2)^2 \qquad (III.17)$$

Ou $x \in [-10^3, 10^3]$ avec n= 1.

III.3.7 KUR :

Le front de Pareto de ce problème teste est un front non-convexe voilà la formule ci-dessous [24]

$$f_1(x) = \sum_{i=1}^{n-1} (-10 \exp(-0.2\sqrt{x_i^2 + x_{i+1}^2})) \qquad (III.18)$$

$$f_2(x) = \sum_{i=1}^{n} (|xi|^{0.8} + 5 \sin x_i^3) \qquad (III.19)$$

Ou $x_i \in [-5, 5]$ pour tout i = 1, . . ., n et n = 3.

III.3.8 QV :

Les composants de la fonction QV sont 2 fonctions multimodales, ici on utilise la version de (Quagliarella and Vicini 1997) voilà la formule ci-dessous [18]

$$f_1(x) = (\frac{1}{n}\sum_{i=1}^{n}(x_i^2 - 10\ cos\ (2\pi\ x_i) + 10))^{\frac{1}{4}} \qquad (III.20)$$

$$f_2(x) = (\frac{1}{n}\sum_{i=1}^{n}((x_i - 1.5)^2 - 10\ cos\ (2\pi\ (x_i - 1.5)) + 10))^{\frac{1}{4}}$$

Ou $x_i \in [-5, 5]$

III.4. Les Mesures de performance

Les deux buts principaux des Algorithmes Evolutionnaires Multi-Objectif sont la convergence vers la surface de Pareto et la distribution uniforme des solutions le long de cette surface. Ces deux aspects doivent donc être pris en compte par des métriques destinées à mesurer la qualité des résultats obtenus par ces algorithmes.

Pour les méthodes d'optimisation stochastiques, aucune garantie de qualité des résultats ne peut en général être donnée à priori. En effet, la notion même de performance expérimentale de ces approches doit tenir compte de leur nature stochastique, qui oblige de se baser sur une analyse statistique de séries d'expériences indépendantes.

De nombreuses mesures quantitatives ont été proposées pour évaluer la qualité de l'ensemble des compromis optimaux. Nous distinguons dans ce qui suit des métriques qui exigent la connaissance de la surface de Pareto exacte, dites exactes, et celles qui n'exigent pas cette information, dites aveugles. Notons que seules ces dernières sont utilisables pour tester l'applicabilité de déférents algorithmes à un problème réelle [1].

III.4.1 Métriques exactes

Sous ce titre, nous réunissons des métriques qui nécessitent la connaissance de la surface de Pareto exacte. Dans ce qui suit, Q désignera l'ensemble des solutions non-dominées trouvées par l'algorithme, n le cardinal de l'ensemble Q et P^* l'ensemble de Pareto exact [1].

- **Taux d'erreur (Error Ration)**

Cette métrique mesure la "proximité" des solutions non-dominées trouvées par un algorithme de la surface de Pareto exacte [14]. Elle est dénie par :

$$ER(Q) = \frac{\sum_{i=1}^{n} e_i}{n} \qquad (III.21)$$

Ou $e_i = 0$ si l'individu $p_i \in P^*$ et $e_i = 1$ sinon. Plus petite est la valeur de ER(Q), meilleur est l'ensemble Q. L'inconvénient de cette métrique est qu'elle ne distingue pas la proximité relative entre P^* et les solutions de Q n'appartenant pas à P^* .

Pour contourner ce problème, on peut introduire un certain seuil δ et redéfinir e_i comme suit [36]

$$e_i = \begin{cases} 0 & si \quad d_i < \delta \\ 1 & sinon \end{cases} \qquad (III.22)$$

Où d_i est la distance euclidienne (mesurée dans l'espace des critères) entre l'individu p_i et P^*

- **Distance générationnelle (Generational Distance)**

Comme l'Error Ratio, la présente métrique est une métrique de convergence. Elle est dénie par [1] :

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{2} \qquad (III.23)$$

Ou d_i est la distance euclidienne (mesurée dans l'espace des critères) entre l'individu p_i et P^* Notons que les fonctions objectives doivent être normalisées avant de calculer les valeurs d_i

III.4.2 Métriques aveugles

Sous ce titre, nous réunissons des métriques qui ne nécessitent pas la connaissance de la surface de Pareto exacte. Ces métriques peuvent être utilisées pour tester l'applicabilité de différents algorithmes à un problème réelle. Notons que la dernière métrique citée « Set Coverage metric », nécessite la disposition d'un ensemble référence de solutions [1].

- **Espacement (Spacing)**

La métrique suivante évalue l'uniformité de la distribution des solutions non-dominées [21] :

$$SP = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - \bar{d})^2} \qquad (III.24)$$

Où d_i défini comme suit :

$$d_i = \min_{\substack{k \in Q \\ k \neq i}} \sum_{m=1}^{M} |f_m^i - f_m^k| \qquad (III.25)$$

Ou $\bar{d}$ est la moyenne des distances d i pour i = 1, . . ., n. SP représente donc la déviation standard des valeurs de distance entre deux solutions consécutives de l'ensemble Q. Plus SP est petit, mieux distribuées sont les solutions. D'ailleurs, une valeur nulle de SP indique que les solutions sont équidistantes. La présente métrique est une métrique de diversité.

- **Comparaison de deux ensembles (Set CoverageMetric)**

Cette métrique est destinée à être utilisée pour comparer deux ensembles non-dominés au sens de Pareto [17]. Elle définit une semi-distance entre deux ensembles des solutions A et B par :

$$C(A,B) = \frac{|\{b \in B \mid \exists a \in A : a \leq b\}|}{|B|} \qquad (III.26)$$

Cette quantité correspond à la proportion des solutions de l'ensemble B, qui sont faiblement dominées par au moins une solution de A. Notons que C (A, B) n'est pas nécessairement égale à 1 – C (B, A). Pour comparer les ensembles A et B, toutes les deux valeurs C (A, B) et C (B, A) doivent être calculées.

Chapitre IV

Simulation et comparaison

IV.1. Introduction

Après avoir définies les outilles nécessaires (voire chapitre III) à notre étude, nous passons à travers ce chapitre à la présentation, comparaison et interprétation des différents résultats obtenus au cours des différentes simulations réalisées. Ce chapitre est structuré de la manière suivante :

- Application de l'algorithme LMOGA sur les problèmes testes (8 fonctions) en utilisant deux types d'opérateurs de variation. Cela nous permettra d'une part de voire le comportement de cet algorithme par rapport à ses paramètres de variation et d'autre part d'obtenir les solutions lexico-max-Ordering qui seront utilisées en terme de connaissance dans l'étape qui suit.

- Dans une première phase, on applique sur nos problèmes testes les algorithmes évolutionnaires NSGAII et SPEAII et dans une seconde phase en introduit la connaissance (sur le front Pareto) fournit par LMOGA. Cela nous permettra, en calculons les différentes métrique de performance, d'étudier l'apport de cette incorporation en terme de convergence et de diversité des solutions trouvées sur le front de Pareto.

IV.2. Application de l'algorithme LMOGA

IV.2.1. Simulation

Dans cette section nous illustrons des échantillons de résultats représentatifs de l'algorithme LMOGA sur les problèmes testes ZDT1 , ZDT2, ZDT3, ZDT4 et ZDT6.

IV.2.1.1. Condition expérimentale

Pour valider et généraliser l'algorithme LMOGA[7] par rapport au nombre de variables n dans les fonctions testes et par rapports aux paramètres de variation, nous avons choisi d'augmenter n et de tester deux opérateurs de variation. Dans [7], les paramètres sont

- individu : chaque Individu est représenté par un vecteur réelle de taille n=2
- Taille de la population : 100 Individus
- Nombre d'évaluation : 250

- Probabilité de croisement : 1
- Probabilité de mutation : 0.01
- Variation usuelle

En ce qui nous concerne nous avons lancé l'algorithme avec les paramètres suivants :

- individu : chaque Individu est représenté par un vecteur réelle de taille n=30
- Taille de la population : 100 Individus
- Test d'arrêt : la stabilisation du solution LMO après un certain nombre d'évaluation tout dépend de la difficulté du problème test
- Probabilité de croisement : 1
- Probabilité de mutation : 0.01
- Variation usuelle/ Variation SBX de DEB[25]

Pour s'assurer de l'unicité de la solution dans l'espace de décision nous avons procédé comme suite : Après les occurrences successives du même point LMO on garde les vecteurs de décision dont l'image est la solution LMO dans une liste pour vérification.

IV.2.1.1.1 Problème ZDT1

On illustre dans la **Figure** IV.1 l'évolution de l'algorithme LMOGA au cours de la résolution de problème ZDT1 avec le teste d'arrêt : 50 occurrences successive du même point

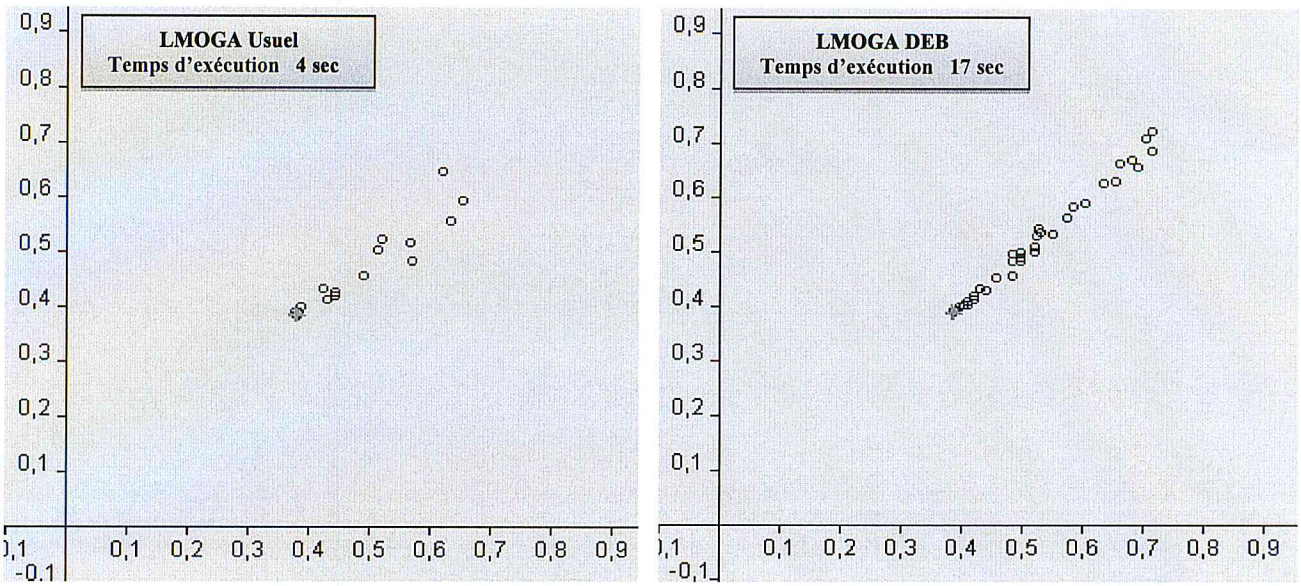


Figure IV.1 évolution de LMOGA lors de la résolution de problème ZDT1dans l'espace des objectifs par rapport aux deux opérateurs de variation

D'après la figure IV.1 on remarque que l'algorithme LMOGA avec la variation usuelle trouve la solution LMO plus rapidement qu'avec la variation de Deb .Les images des solutions LMO trouvées par les 2 expériences :

LMOGA avec variation usuelle	LMOGA avec variation DEB
F1(X) =0.3832	F1(X) =0.3876
F2(X) =0.3831	F2(X) =0.3863

On constate qu'à travers ces valeurs que la solution X trouvée en utilisant la variation usuelle est meilleure et nous confirment qu'elle est unique :

X= {0.3832, 0.0001, 0.0018, 0.0015, 0.0001, 0.0008, 0.0, 0.0001, 0.0, 0.0036,

0.0, 0.0, 0.0019, 0.0001 ,0.0, 0.0008, 0.0, 0.0002, 0.0005, 0.0015,

0.000 3, 0.0, 0.0007, 0.0002, 0.0001, 0.0009, 0.0001, 0.0042, 0.0, 0.0}

IV.2.1.1.2 Problème ZDT2

On illustre dans la **Figure** IV. 2 l'évolution de l'algorithme LMOGA au cours de résolution de problème ZDT2 avec le teste d'arrêt : 100 occurrences successive du même point

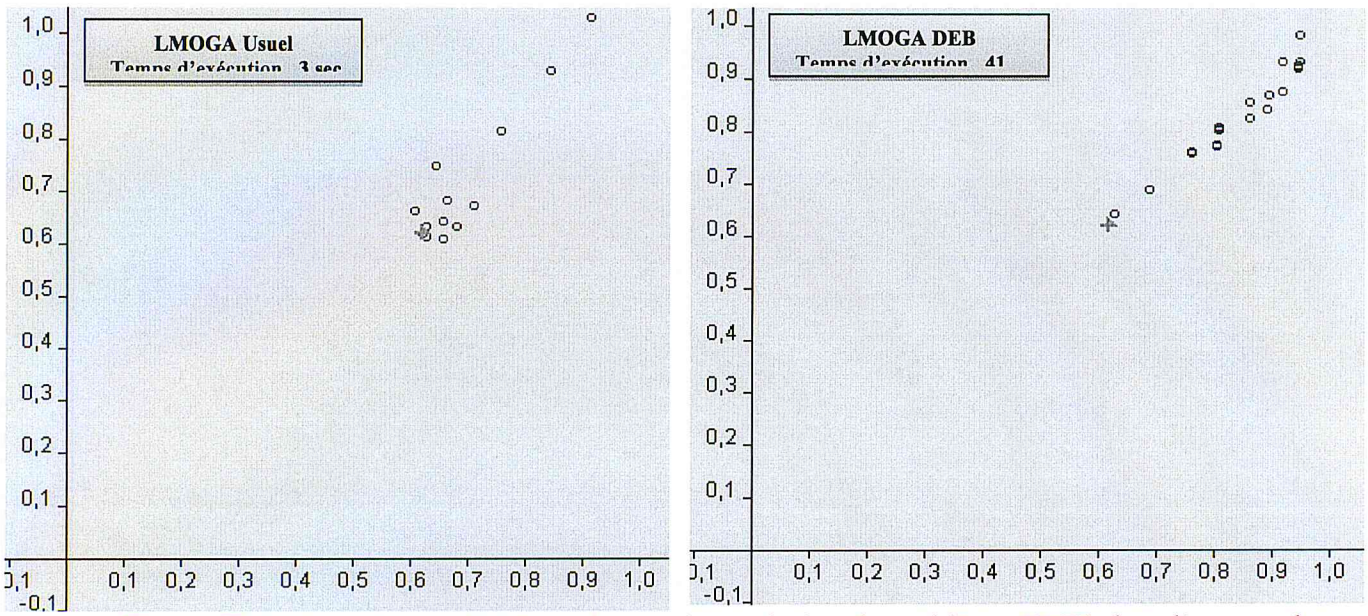


Figure IV.2 évolution de LMOGA lors de la résolution de problème ZDT2 dans l'espace des objectifs par rapport aux deux opérateurs de variation

D'après la figure IV.2 on remarque que l'algorithme LMOGA avec la variation usuelle trouve la solution LMO plus rapidement qu'avec la variation Deb. Les images des solutions LMO trouvées au cours des 2 simulations sont :

LMOGA avec variation usuelle	LMOGA avec variation DEB
F1(X) = 0.6188	F1(X) =0.6185
F2(X) =0.6185	F2(X) =0.6175

On constate que la solution LMO trouvé par LMOGA avec la variation de DEB est meilleure, elle est unique dans l'espace de décision dont sa valeur est :

X= { 0.6185, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0002, 0.0, 0.0, 0.0001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}

IV.2.1.1.3 Problème ZDT3

On illustre dans la **Figure** IV. 3 l'évolution de l'algorithme LMOGA au cours de résolution de problème ZDT3, après 50 occurrences successif du même point

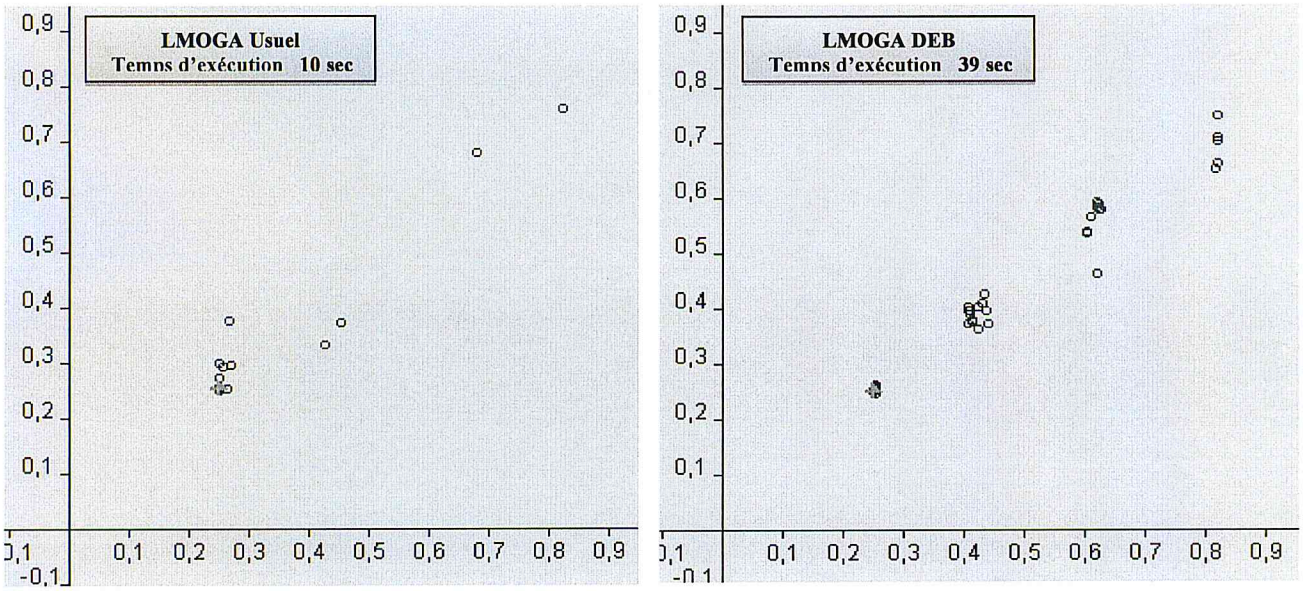


Figure IV.3 évolution de LMOGA lors de la résolution de problème ZDT3 dans l'espace des objectifs par rapport aux deux opérateurs de variation

D'après la figure IV.3 on remarque que l'algorithme LMOGA avec la variation usuelle trouve la solution LMO plus rapidement qu'avec la variation Deb. Les images des solutions LMO trouvées au cours des 2 simulations sont :

LMOGA avec variation usuelle	LMOGA avec variation DEB
F1(X) =0. 2501	F1(X) =0.2500
F2(X) = 0.2500	F2(X) =0.2500

On constate que la solution LMO trouvé par LMOGA avec la variation de DEB est meilleure, elle est unique dans l'espace de décision dont sa valeur est :

X ={ 0.25,0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}

IV.2.1.1.4 Problème ZDT4

On illustre dans la **Figure** IV. 4 l'évolution de l'algorithme LMOGA au cours de la résolution du problème ZDT4, après 100 occurrences successif du même vecteur de décision

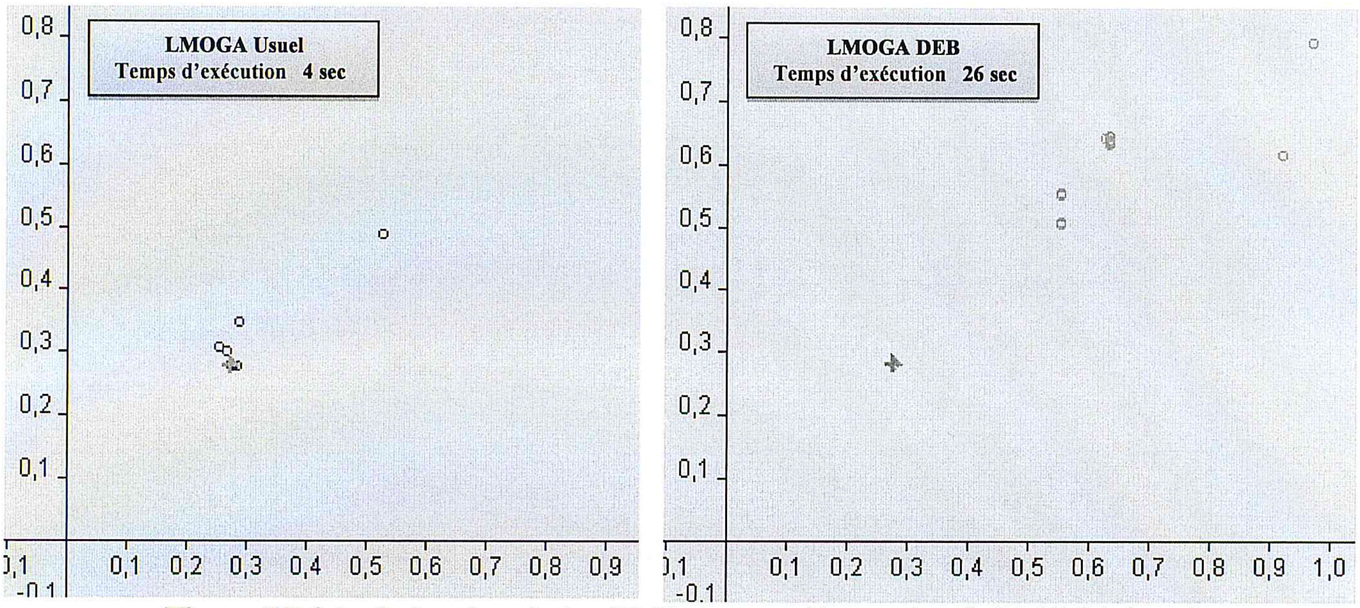


Figure IV.4:évolution du solution LMO au cours l'execution de LMOGA lors de résolution de problème ZDT4

D'après la figure IV.4 on remarque que l'algorithme LMOGA avec la variation usuelle trouve la solution LMO plus rapidement qu'avec la variation Deb. Les images des solutions LMO trouvées au cours des 2 simulations sont :

LMOGA avec variation usuelle	LMOGA avec variation DEB
F1(X) =0.2752	F1(X) =0. 2768
F2(X) = 0.2756	F2(X) =0.2789

On constate qu'à travers ces valeurs que la solution X trouvée en utilisant la variation usuelle est meilleure et nous confirment qu'elle est unique :

X ={0.2752 ,0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}

IV.2.1.1.5 Problème ZDT6

On illustre dans la **Figure** IV. 5 l'évolution de l'algorithme LMOGA au cours de la résolution du problème ZDT6, après 100 occurrences successif du même point.

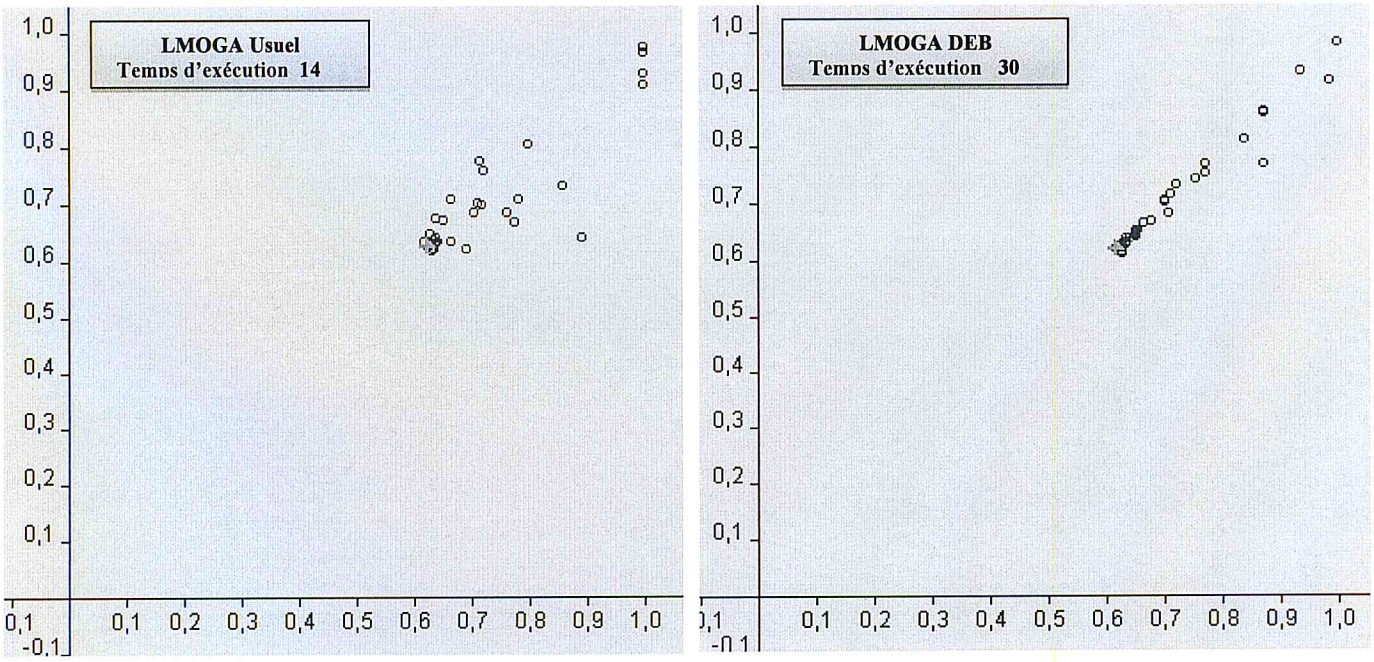


Figure IV.5 évolution de LMOGA lors de la résolution de problème ZDT6 dans l'espace des objectifs par rapport aux deux opérateurs de variation

D'après la figure IV.5 on remarque que l'algorithme LMOGA avec la variation usuelle trouve la solution LMO plus rapidement qu'avec la variation de Deb. Les images des solutions LMO trouvées au cours des 2 simulations sont :

LMOGA avec variation usuelle	LMOGA avec variation DEB
F1(X) = 0.6162	F1(X) =0. 6189
F2(X) = 0.6139	F2(X) =0. 6187

On constate que la solution trouvée par LMOGA avec la variation usuelle est mieux et le vecteur X est unique et dont sa valeur est :

X= {0.0892, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}

Remarque :

Les solutions présentées précédemment sont les valeurs retenues comme connaissance à incorporer dans l'initialisation des algorithmes multi objectifs considérés dans cette étude. Concernant l'évaluation de l'algorithme LMOGA par rapport aux deux opérateurs de variation, nous avons effectué 50 simulations pour chaque opérateur par fonction et nous avons procéder aux comparaisons suivante.

IV.2.2 Comparaison

Notre outil de simulation et de comparaison nous permet d'effectuer des comparaisons en utilisant des tableaux. Les tableaux ci-dessous représentent les résultats de LMOGA pour 50 populations initiales différentes pour les fonctions testes : ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, SCH, KUR et QV.

Tableau IV.1 : comparaison entre LMOGA avec variation usuelle et LMOGA avec variation Deb en termes du nombre d'itérations pour 50 initialisations différentes

Problèmes Test	Nombre d'itération							
	Minimum		Maximum		Moyenne		Ecart type	
	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB
ZDT1	175	2355	420	5012	298.0	4023.0	73.36	877.34
ZDT2	148	926	485	3026	264.0	1725.0	85.22	550.77
ZDT3	149	1642	544	4523	340.0	3877.0	108.70	764.87
ZDT4	122	835	406	2876	224.0	1689.0	73.44	552.78
ZDT6	120	688	529	3528	294.0	2386.0	107.03	826.60
SCH	102	781	370	1713	179.0	1307.0	77.22	226.16
KUR	122	301	341	498	194.0	314.0	56.55	35.10
QV	112	584	357	2441	192.0	1198.0	71.62	416.46

Tableau IV.2 : comparaison entre LMOGA avec variation usuelle et LMOGA avec variation DEB en termes du temps d'exécution 50 initialisations différentes

Problèmes Test	Temps d'exécution (S)							
	Minimum		Maximum		Moyenne		Ecart type	
	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB	LMOGA NORMAL	LMOGA DEB
ZDT1	2.574	26.006	6.521	56.098	4.357	42.975	1.112	9.762
ZDT2	2.215	12.386	6.817	40.388	3.901	23.536	1.192	7.262
ZDT3	2.184	18.252	7.738	45.895	4.901	38.309	1.520	6.309
ZDT4	1.669	9.86	5.195	31.091	2.987	19.657	0.935	6.297
ZDT6	1.622	6.77	7.16	29.64	4.015	20.506	1.347	6.587
SCH	0.967	5.163	3.619	10.67	1.877	8.257	0.797	1.31
KUR	1.591	1.903	3.555	5.136	2.655	3.4336	0.748	0.276
QV	1.529	4.477	4.821	18.346	2.627	8.925	0.925	3.048

A partir du Tableau IV.1 nous remarquons que LMOGA avec la variation usuelle trouve la solution LMO avec un nombre d'itérations plus inférieur que le nombre d'itérations effectuées par LMOGA avec variation de Deb et cela pour tous les problèmes test : ZDT1, ZDT2, ZDT3 ZDT4, ZDT6, KUR, SCH, QV.

A partir du Tableau IV.2 nous remarquons que LMOGA avec la variation normale trouve la solution LMO en temps plus réduit que le temps effectués par LMOGA avec la variation de Deb pour tous les problèmes test : ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, SCH, KUR et QV.

Donc la variation usuelle est la plus efficace pour l'algorithme LMOGA en termes de vitesse de convergence, et en termes de nombre d'évaluations dans l'algorithme (itérations).

Remarque importante :

Les valeurs des solutions obtenues dépendent de l'opérateur de variation et du teste d'arrêt (nombre d'itérations sans changement), cela veut dire que les solutions sont approximatives d'où la nécessité de penser à « l'incorporation de connaissances approximatives ou bruités ».

IV.3. NSGAII et SPEAII

IV.3.1. Simulation

Notre outil de simulation permet d'effectuer plusieurs simulation et comparaison des algorithmes NSGAII et SPEAII, avec les différent problèmes tests ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, SCH, KUR et QV.

IV.3.1.1. Résultats expérimentaux

Dans cette section on illustre des échantillons de résultats représentatifs des deux algorithmes NSGAII et SPEAII, lors de la résolution des problèmes test ZDT1, ZDT2, ZDT3, ZDT4 ZDT6 et QV

IV.3.1.1.1 Conditions expérimentales

Pour qu'on puisse voire l'impact de l'incorporation de la solution LMO sur chacun des deux algorithmes, (NSGAII et SPEAII), et pour pouvoir comparer les résultats relatifs à chaque algorithme avec ceux de l'autres , nous avons choisi d'utiliser exactement les mêmes paramètres utilisés par DEB[1] pour l'algorithme NSGAII, ainsi que ceux utilisés par ZIZTLER [18] pour valider l'algorithme SPEAII. Ainsi nous avons lancé les deux algorithmes simultanément avec et sans l'incorporation de la solution LMO dans la population initiale et avec les paramètres suivantes:

- Taille de la population : 100
- Probabilité de croisement : 1
- Probabilité de mutation : 0.01
- Opérateur de variation : croisement SBX+ mutation polynomiale

Voici un échantillon des résultats de simulation pour cette expérience avec certain problème tests :

IV.3.1.1.2 problème ZDT1

La figure (IV.6) représente l'évolution de NSGAII lors de la résolution du problème ZDT1, elle représente les deux surfaces de compromis trouvées par NSGAII avec et sans l'incorporation de la solution LEX-MAX –ORDERING (LMO), après 250 itérations. De même La figure (IV.7) représente l'évolution de SPEAII lors de la résolution du même problème ZDT1, après 500 itérations

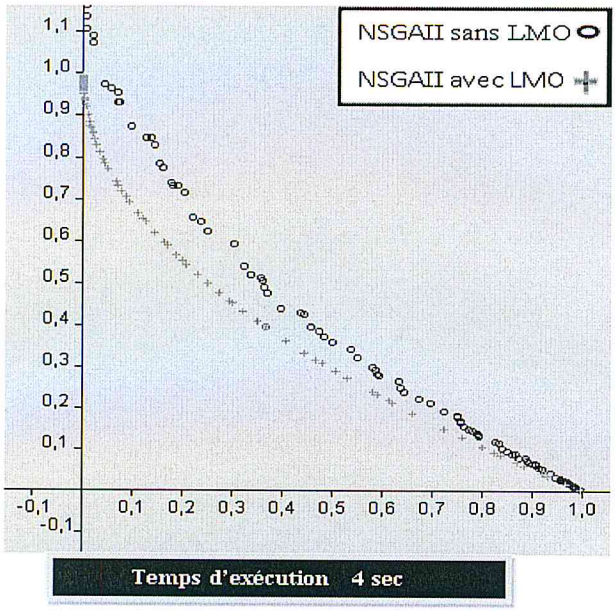


Figure IV.6 : évolution de NSGAII lors de résolution du problème ZDT1

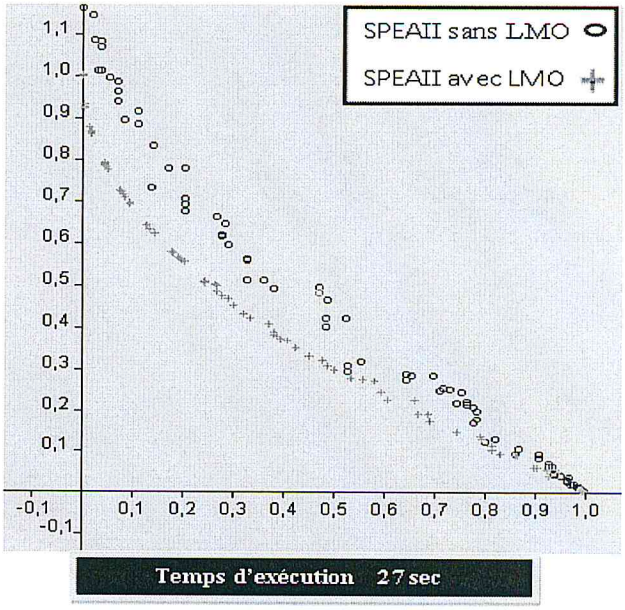


Figure IV.7 : évolution de SPEAII lors de résolution du problème ZDT1

L'effet de la solution LMO est très positif sur les deux algorithmes et cela en provoquant une accélération de convergence. De plus selon les figures IV.6 et IV.7nous remarquons que NSGAII converge plus rapidement vers le front de Pareto que SPEAII mais n'oubliant pas qu'on on est en face d'un problème test facile et rapide en terme de convergence cela est dû à la nature de problème ZDT1.

IV.3.1.1.3 problème ZDT2

La figure (IV.8) représente l'évolution, (après 300 itérations) de NSGAII lors de la résolution du problème ZDT2. La figure (IV.9) représente l'évolution (après1000 itérations) de SPEAII lors de la résolution du même problème.

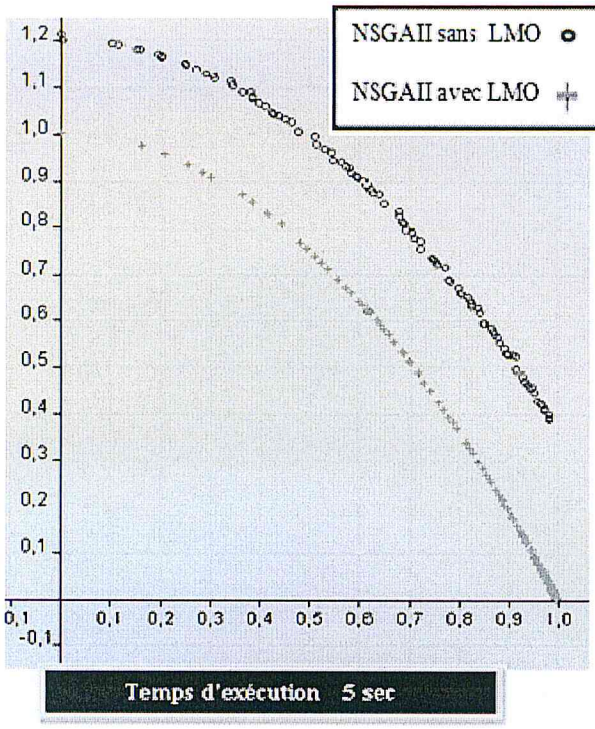


Figure IV.8 : évolution de NSGAII lors de résolution du problème ZDT2

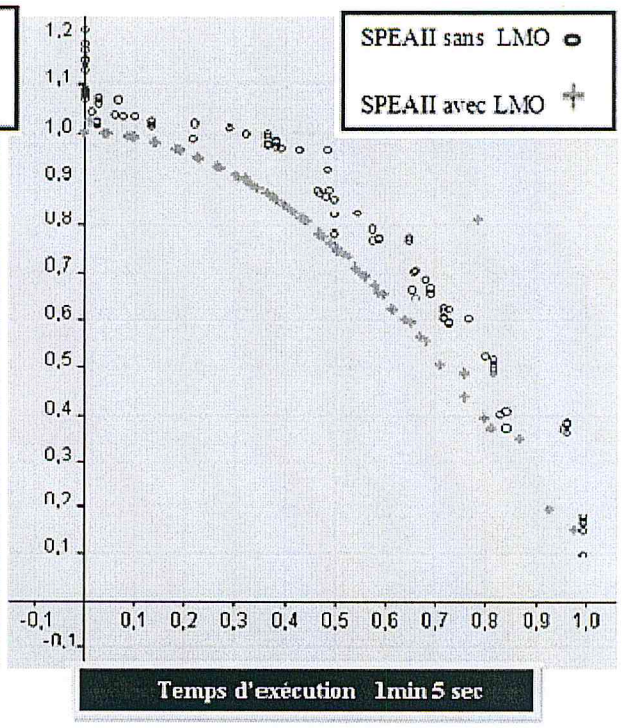


Figure IV.9 : évolution de SPEAII lors de résolution du problème ZDT2

L'effet de la solution LMO est très flagrant pour NSGAII et moins important pour SPEAII. La convergence des deux algorithmes sans l'incorporation de la solution LMO est encore très lente, cela est peut-être dû à la nature du problème traité.

IV.2.1.1.4 problème ZDT3

La figure (IV.10) représente l'évolution (après 300 itérations) de NSGAII lors de la résolution du problème ZDT3, La figure (IV.11) représente l'évolution (après 1000 itérations) de SPEAII lors de la résolution du même problème.

L'effet de la solution LMO est moins flagrant pour ce problème et plus particulièrement pour NSGAII. Malgré que ce dernier soit plus rapide que SPEAII.

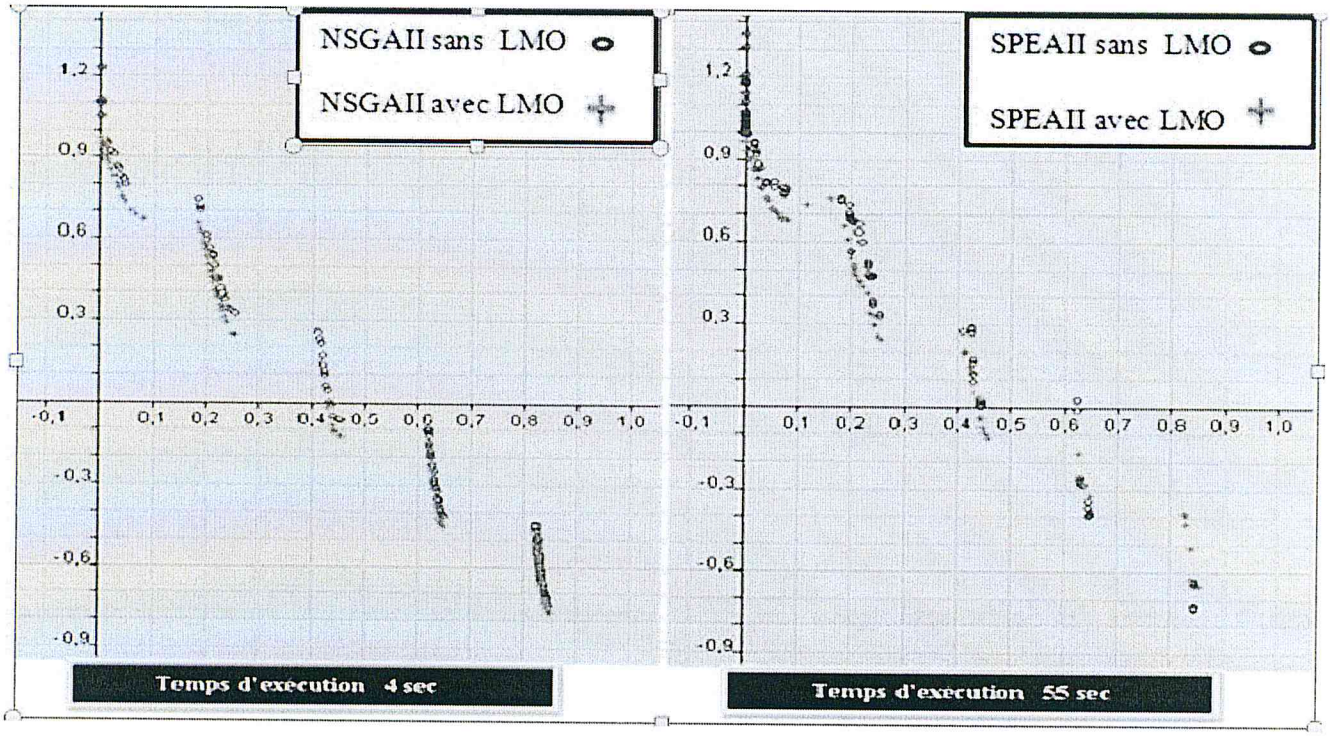


Figure IV.10 évolution de NSGAII lors de résolution du problème ZDT3

Figure IV.11 évolution de SPEAII lors de résolution du problème ZDT3

IV.3.1.1.5 problème ZDT4

La figure (IV.12) représente l'évolution (après 300 itérations) de NSGAII lors de la résolution du problème ZDT4 et la figure (IV.13) représente l'évolution (après1000 itérations) de SPEAII lors de la résolution du même problème.

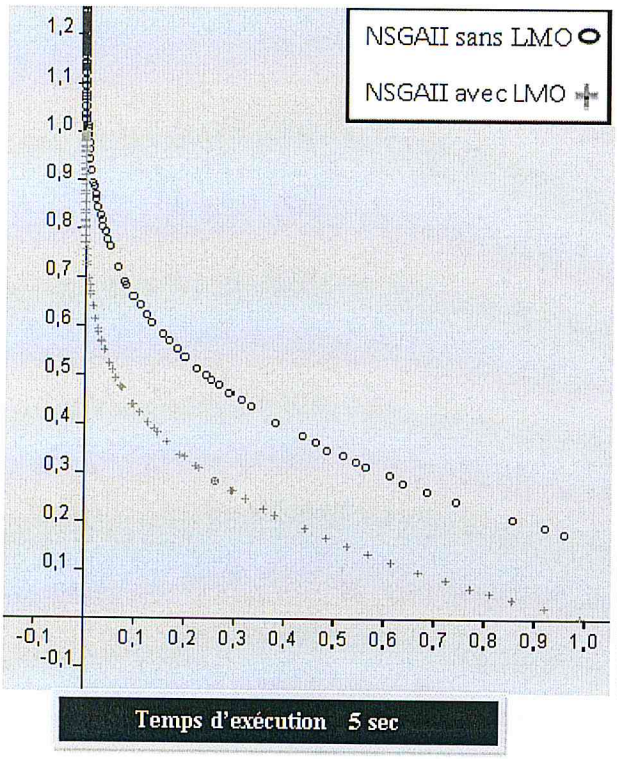


Figure IV.12 : évolution de NSGAII lors de résolution du problème ZDT4

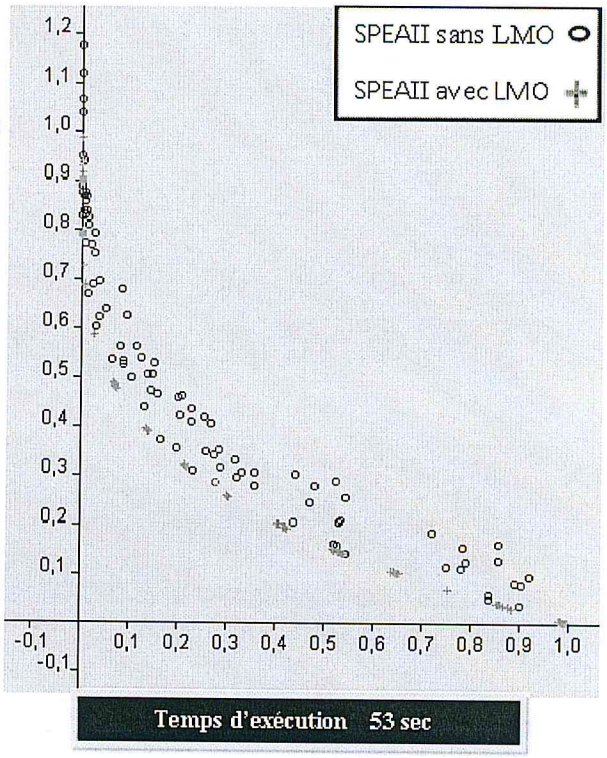


Figure IV.13 : évolution de SPEAII de résolution du problème ZDT4

Le bienfait de la solution LMO est très visible et plus particulièrement pour NSGAII. On remarque aussi qu'il y a une différence importante en termes de temps d'exécution entre NSGAII et SPEAII.

IV.3.1.1.6 problème ZDT6

La figure (IV.14) représente l'évolution (après 500 itérations) de NSGAII lors de la résolution du problème ZDT6 et la figure (IV.15) représente l'évolution (après 2000 itérations) de SPEAII lors de la résolution du même problème.

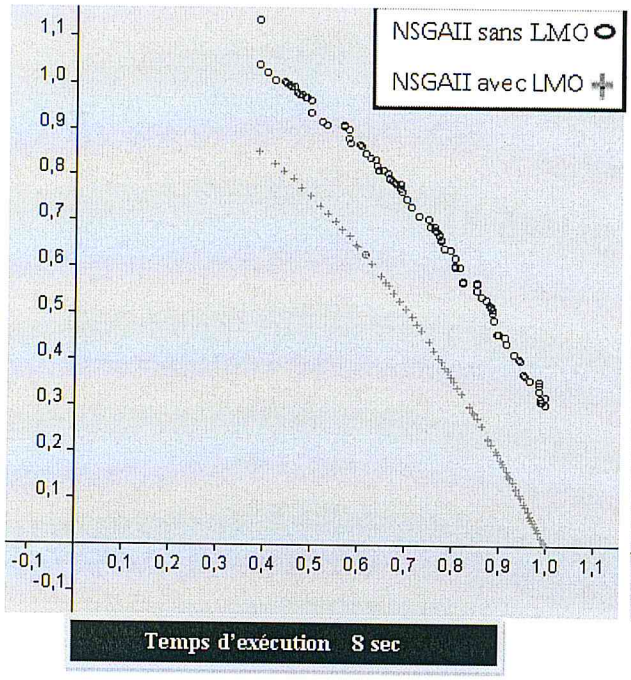


Figure IV.14 : évolution de NSGAII lors de résolution du problème ZDT6

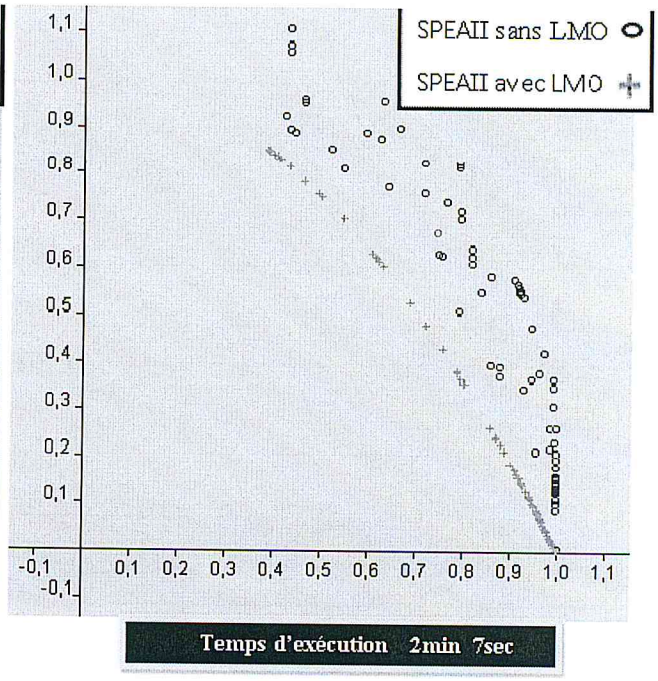


Figure IV.15 : évolution de SPEAII lors de résolution du problème ZDT6

L'apport de la solution LMO est très visible dans les deux simulations présentées. Nous constatons aussi que les deux Algorithmes n'arrivent pas à couvrir l'intervalle [0, 0.4] du front de Pareto, c'est dû en fait à la difficulté du problème test ZDT6

IV.3.1.1.7 problème QV

La figure (IV.16) représente l'évolution (après 50 itérations) de NSGAII lors de la résolution du problème QV et La figure (IV.17) représente l'évolution (après 250 itérations) de SPEAII lors de la résolution du même problème.

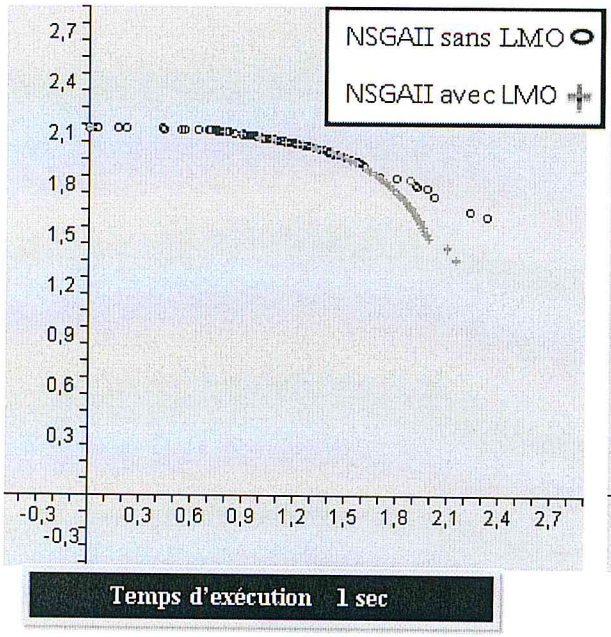


Figure IV.16 : évolution de NSGAII lors de résolution du problème QV

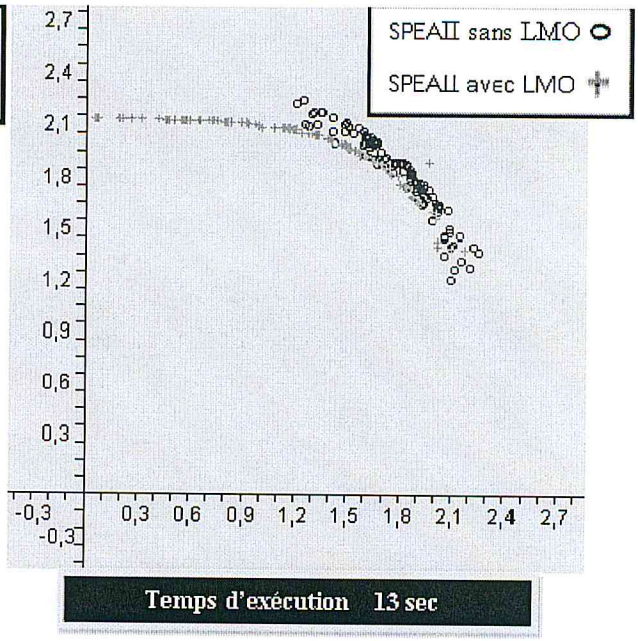


Figure IV.17 : évolution de SPEAII de résolution du problème QV

Le bienfait de la solution LMO est assez visible sur les deux figures et plus particulièrement pour SPEAII.

IV.3.2. Comparaison

Afin de confirmer numériquement les résultats mentionnés à travers les analyses précédentes, nous listons dans les figures suivantes les valeurs moyennes des métriques : taux d'erreur (ER) Generational Distance (GD) Set CoverageMetric (SCM) Espacement (Spacing) et temps d'exécution par rapport aux nombre de génération prises sur des essais et pour cela nous avons choisi le problème ZDT1.

IV.3.2.1 problème ZDT1

La figure (IV.18) représente l'évolution du taux d'erreur en fonction du nombre d'itération lors de la résolution du problème ZDT1 pour l'algorithme NSGAIIL. La figure (IV.19) représente l'évolution du taux d'erreur en fonction du nombre d'itération lors de la résolution du problème ZDT1 pour l'algorithme SPEAII

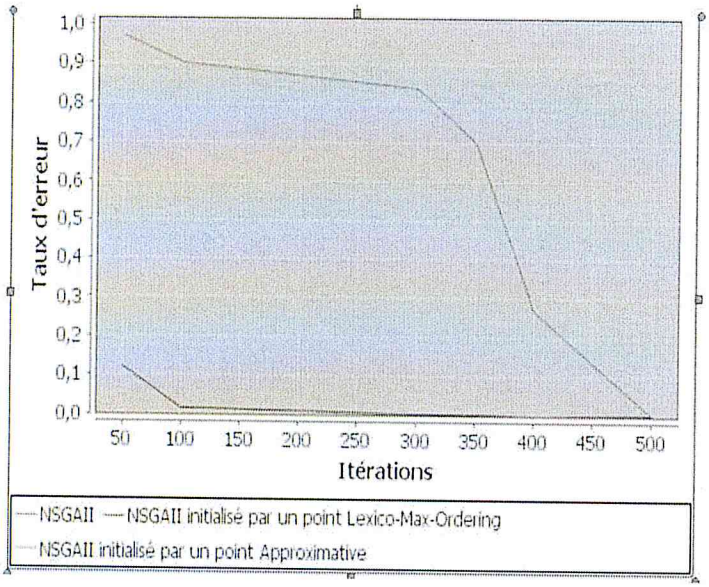


Figure IV.18 : NSGAII : évolution du taux d'erreur lors la résolution ZDT1

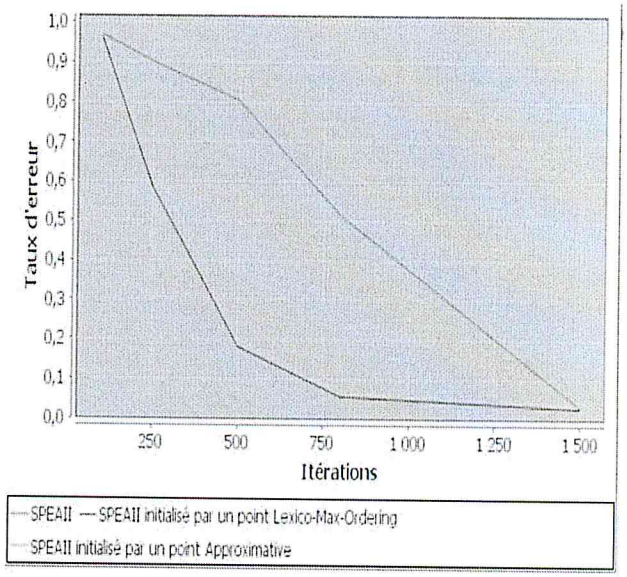


Figure IV.19 : SPEAII : évolution du taux d'erreur lors la résolution ZDT1

La figure (IV.20) et la figure (IV.21) représentent respectivement l'évolution de la distance générationnelle en fonction de nombre d'itération des algorithmes NSGAII et SPEAII

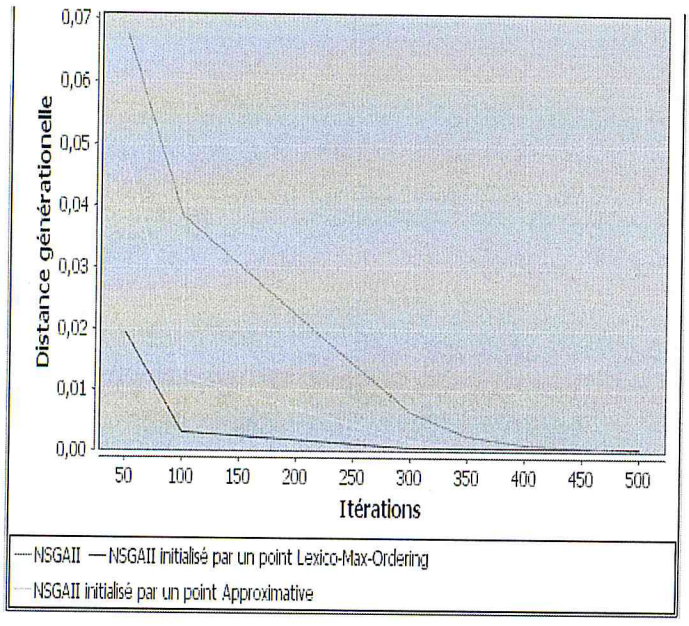


Figure IV.20: NSGAII: évolution de distance distance générationnelle lors la résolution ZDT1 résolution ZDT1

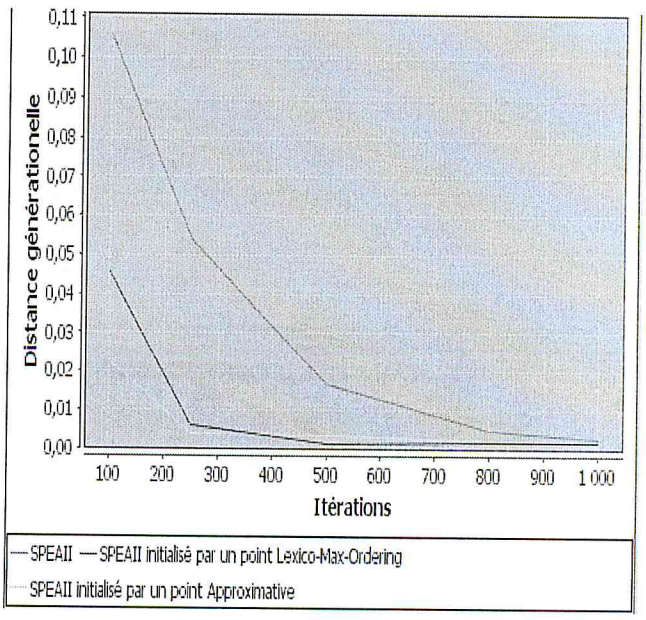


Figure IV.21 : SPEAII : évolution de distance générationnelle lors la

La figure (IV.22) et la figure(IV.23) représentent respectivement l'évolution de l'espacement en fonction du nombre d'itération respectivement pour les algorithmes NSGAII et SPEAII.

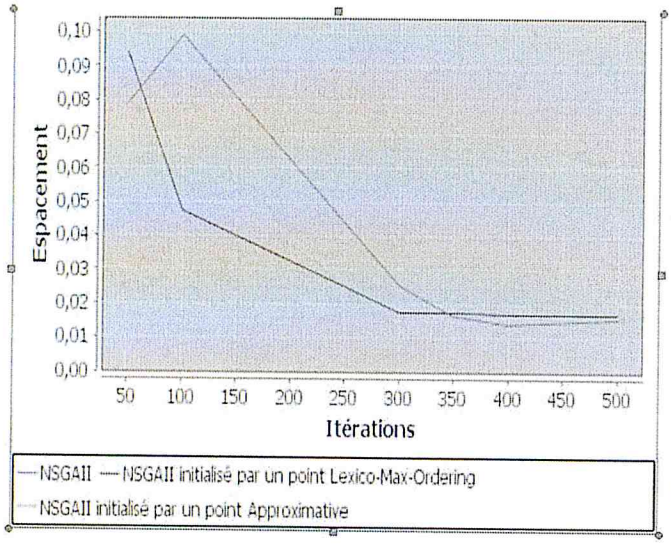


Figure IV.22 : NSGAII : évolution d'espacement lors la résolution ZDT1

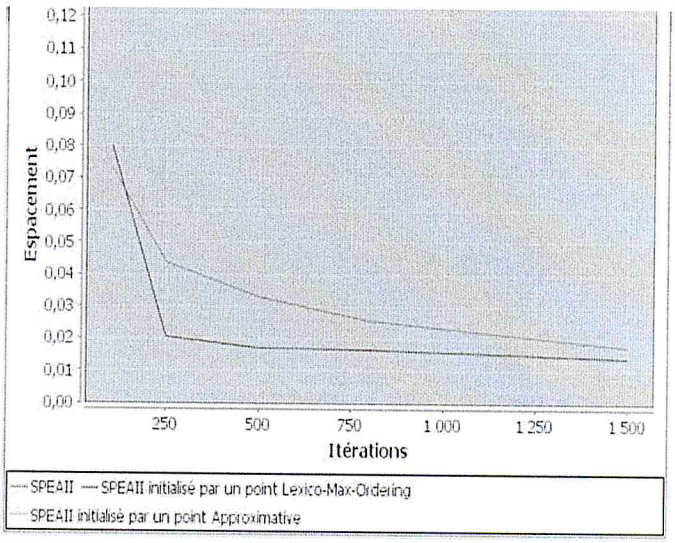


Figure IV.23 : SPEAII : évolution d'espacement lors la résolution ZDT1

La figure (IV.24) et la figure (IV.25) représentent respectivement l'évolution de la mesure de SCM(Set Coverage Metric) en fonction de nombre d'itération respectivement pour les algorithmes NSGAII et SPEA II

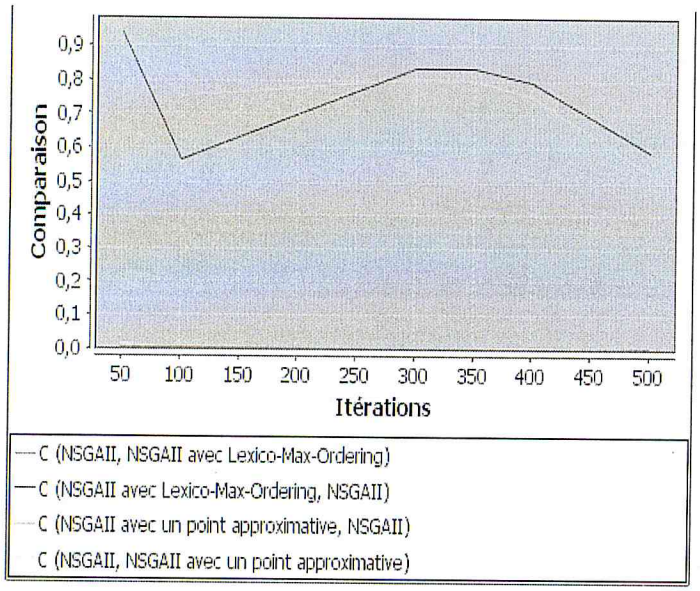


Figure IV.24 :NSGAII: évolution de SCM lors de la résolution ZDT1

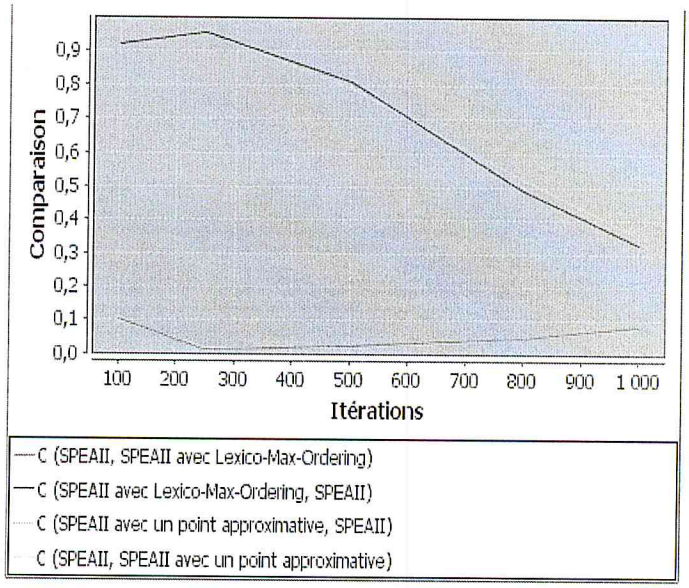


Figure IV.25 : SPEAII: évolution de SCM lors de la résolution ZDT1

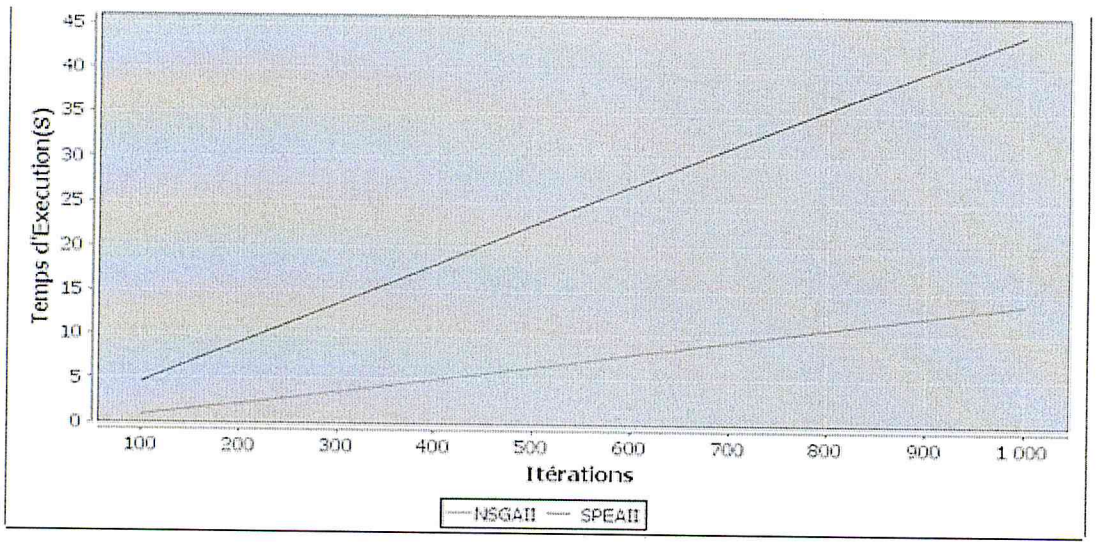


Figure IV.26 : évolution du temps en fonction de nombre d'itérations pour NSGAII et SPEAII

La figure IV.26 montre que SPEAII consomme plus de temps que NSGAII dans les itérations cela est du surement à la procédure du clustering (utilisée dans SPEAII) qu'est plus lourde que son équivalent (crowding) dans NSGEAII.

On illustre dans Le tableau IV.3 les valeurs des différentes métriques (calculées à 2000 itérations) pour les deux algorithmes sur les 8 problèmes testes prés en considération avec : l'incorporation des solutions LMO, l'incorporation d'une solution LMO approximative (bruité) et sans incorporation.

Tableau IV.3 : comparaison de performance de NSGAII ,SPEAII sans et avec l'incorporation du solution LMO exacte ou approximative après 2000 itérations.

Probbleme teste	MOEA	Taux d'erreur	Distance générati...	Espacement	C (A, B)	C (A, C)	Temps d'execution ...
ZDT1	A: NSGA II ...	0,000	0,000	0,015	0,000	0,030	46,423
	B: NSGAII avec M...	0,000	0,000	0,018	0,010	0,030	32,741
	C: NSGAII avec P...	0,000	0,000	0,018	0,000	0,000	46,695
	A: SPEA II ...	0,010	0,000	0,006	0,040	0,110	148,854
	B: SPEAII avec Ma...	0,010	0,001	0,017	0,060	0,100	107,016
	C: SPEAII avec Po...	0,020	0,001	0,014	0,040	0,030	139,236
ZDT2	A: NSGA II ...	0,030	0,000	0,010	0,000	0,030	66,482
	B: NSGAII avec M...	0,030	0,000	0,007	0,010	0,080	46,623
	C: NSGAII avec P...	0,080	0,001	0,007	0,010	0,010	65,325
	A: SPEA II ...	0,110	0,001	0,017	0,010	0,020	187,724
	B: SPEAII avec Ma...	0,080	0,001	0,008	0,240	0,160	119,650
	C: SPEAII avec Po...	0,070	0,001	0,021	0,250	0,010	163,529
ZDT3	A: NSGA II ...	0,020	0,000	0,009	0,000	0,000	38,442
	B: NSGAII avec M...	0,000	0,000	0,010	0,010	0,010	26,539
	C: NSGAII avec P...	0,000	0,000	0,010	0,050	0,010	39,124
	A: SPEA II ...	0,030	0,000	0,022	0,100	0,180	148,399
	B: SPEAII avec Ma...	0,050	0,001	0,019	0,160	0,140	106,700
	C: SPEAII avec Po...	0,070	0,003	0,031	0,160	0,160	126,946
ZDT4	A: NSGA II ...	0,350	0,001	0,020	0,000	0,000	48,368
	B: NSGAII avec M...	0,000	0,000	0,017	0,760	0,750	31,961
	C: NSGAII avec P...	0,200	0,001	0,019	0,300	0,000	49,035
	A: SPEA II ...	0,020	1,166	11,608	0,000	0,010	183,042
	B: SPEAII avec Ma...	0,010	0,216	2,151	0,030	0,010	110,427
	C: SPEAII avec Po...	0,030	0,967	9,627	0,060	0,000	160,110
ZDT6	A: NSGA II ...	0,020	0,001	0,014	0,000	0,000	39,801
	B: NSGAII avec M...	0,000	0,000	0,014	0,360	0,460	28,715
	C: NSGAII avec P...	0,000	0,000	0,011	0,220	0,000	40,943
	A: SPEA II ...	0,960	0,161	0,119	0,060	0,200	209,739
	B: SPEAII avec Ma...	0,080	0,084	0,110	0,670	0,810	133,520
	C: SPEAII avec Po...	0,850	0,043	0,032	0,610	0,050	207,493
SCH	A: NSGA II ...	0,030	0,000	0,032	0,010	0,000	39,765
	B: NSGAII avec M...	0,080	0,001	0,030	0,010	0,000	26,958
	C: NSGAII avec P...	0,090	0,000	0,032	0,000	0,000	39,694
	A: SPEA II ...	0,000	0,000	0,000	0,000	0,000	243,977
	B: SPEAII avec Ma...	0,000	0,000	0,398	1,000	0,000	174,956
	C: SPEAII avec Po...	0,000	0,000	0,068	1,000	0,990	263,113
KUR	A: NSGA II ...	0,600	0,004	0,116	0,060	0,080	50,092
	B: NSGAII avec M...	0,590	0,003	0,104	0,120	0,080	48,407
	C: NSGAII avec P...	0,560	0,004	0,118	0,110	0,070	50,092
	A: SPEA II ...	0,590	0,156	2,144	0,190	0,260	162,412
	B: SPEAII avec Ma...	0,530	0,007	0,175	0,080	0,190	114,551
	C: SPEAII avec Po...	0,550	0,010	0,209	0,080	0,140	151,773
QV	A: NSGA II ...	0,070	0,001	0,094	0,080	0,090	41,652
	B: NSGAII avec M...	0,060	0,000	0,061	0,140	0,050	37,596
	C: NSGAII avec P...	0,030	0,000	0,069	0,180	0,040	38,470
	A: SPEA II ...	0,900	0,005	0,029	0,040	0,030	135,658
	B: SPEAII avec Ma...	0,720	0,007	0,070	0,280	0,120	94,303
	C: SPEAII avec Po...	0,760	0,006	0,047	0,150	0,150	130,900

IV.4. Conclusion :

Les résultats prouvent que l'algorithme LMOGA se porte mieux avec la variation usuelle en termes du nombre d'itération et même en termes de qualité des solutions trouvées. L'application de cet algorithme sur des fonctions tests contenant jusqu'à 30 variables de décision, nous informe d' une part de sa robustesse par rapport à ce paramètre et d'autre part est une généralisation du travail présenté dans [7] (dans le quel l'algorithme est appliqué uniquement sur des problèmes tests à deux variable de décision).

L'incorporation d'une solution LMO dans les populations initiales des deux algorithmes évolutionnaires multi objectifs donne un plus en terme de vitesse de convergence à ces deux algorithmes mais pas forcément en terme de diversité. Nous avons remarqué aux cours des différentes simulations qu'une valeur moins précise (bruitée) de la solution LMO à un effet réduit sur l'accélération de l'algorithme.

Concernant les deux algorithmes considérés dans cette étude et à travers les problèmes teste pris en charge (ZDT1 ZDT2 ZDT3 ZDT4 ZDT6 SCH KUR QV), nous remarquons que NSGAII est plus efficace que SPEAII en terme de vitesse de convergence.

Conclusion générale

Conclusion et prospectives

Dans ce mémoire nous avons abordé et mise en œuvre quelques algorithmes évolutionnaire multi-objectif à fin de mettre en évidence quelques aspects à la recherche.

à partir des expériences effectués, nous avons remarqués que la variation génétique usuelle aide l'approche LMOGA pour trouver la solution LMO plus rapidement que si on utilise la variation Deb[1].

Les résultats obtenus montrent qu'une solution LMO donne de la performance aux algorithmes évolutionnaires quand elle est introduite dans la population initiale. Cette performance on la remarque à travers la vitesse de convergence de l'algorithme mais pas à travers diversité des solutions trouvées sur le front de Pareto.

Nos expériences montrent aussi la rapidité remarquable de NSGAII en termes de qualité de solutions obtenues et de vitesse de convergence vers la surface Pareto optimale par rapport à l'algorithme SPEAII pendant la résolution des problème test Zitzler, Deb &Thiele [10].

Pour mieux cerné cette étude et avoir une vue plus générale, nous exprimons les perspectives suivantes :

- Appliquer cette incorporation de connaissance fournis par LMOGA sur d'autres algorithmes évolutionnaires multi objectifs.

- Utiliser des problèmes testes de dimension supérieure (par exemple, des problèmes tri-objectifs) avec plus de difficultés.

Références Bibliographie

Bibliographie

[1] Ahmed Chamseddine BEN ABDALLAH .Rapport de Mémoire de Master d'IngénierieMathématique. Optimisation multi-objectif évolutionnaire.Ecole P O L Y T E C H N I Q U E D E T U N I S I E,2004/2005

[2] Alain Berro , Olivier Heguy . Agents évolutionnaires pour l'optimisation multiobjectifEvolutionary agents for multiobjectiveoptimization.Article Université Toulouse I Sciences Sociales (UT1), France

[3] Amir NAKIB .Conception de métaheuristique d'optimisation pour la segmentation d'images. Application à des images biomédicale .these de doctorat de l'université paris 12-VAL de Marne ,5 Décembre 2007

[4] Anne Spalanzani .Algorithmes évolutionnaires pour l'étude de la robustesse des systèmes de reconnaissance automatique de la parole .Thèse de doctorat,Université Joseph Fourier, Grenoble ,28 Octobre 1999

[5] B.Ahiod, I. Berrada.I. Kassou .Deux Méthodes de Recherche Locale pour Résoudre un Problème d'Horaire du Personnel Infirmier dans un établissement Hospitalier . 1998

[6] BELBACHIR Assia DEAU Raphaël LENNE Renaud SNOUSSI Jihene .la programmation génétique .rapport de laboratoire des sciences de l'informationnel des system(LSIS).

[7] bouyekhf karima, bouchama imene .algorithmes evolutionnaires multiobjectifs: application à la recherché de l'optimum lexicoraphique max-oredring. Mémoire pour obtenir diplôme master en Informatique, Université Saad Dahleb Blida juillet 2012

[8] C. A. CoelloCoello, D. A. Van Veldhuizen, and G. B. Lamont .Evolutionary Algorithms for Solving Multi-Objective Problems . Kluwer Academic Publishers, 2002.

[9] C.M. Fonseca and P.J. Fleming .Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms .Part 1: A Unified Formulation, IEEE Transactions on Systems,Man and Cybernetics, Vol. 28, No. 1, pp. 26-37, 1998.

[10] C.Z. Janikow and Z. Michalewiz.An Experimental Comparaison of Binary and Floating Point Representations in Genetic Algorithms,.In Proceedings of the 4thInternational Conference on Genetic Algorithms, pp. 31-36, (1991).

[11] D.E. Goldberg . Genetic Algorithms in Search Optimisation and Machine Learning.Addison-Wesley, 1989.

[12] D.E. Goldberg, J. Richardson . Genetic Algorithms with Sharing for Multimodal Function Optimization .Dans Genetic Algorithms and their Applications. Proceedings of the Second International Conference on Genetic Algorithms; pages 41-49; Lawrence Erlbaum; 1987.

[13] Dominique Francisci .Algorithmes évolutionnaires et optimisation multi-objectifs en data mining .rapport de recherche.laboratoire informatique signaux et systemes de Sophia , mars 2002

[14] D. V. Veldhuizen. Multiobjective Evolutionnary Algorithms :Classifications, Analysis and Innovations. PhD thesis, Air Force Institute of Technology, Dayton,1999.

[15] D. Whitley. A Free Lunch Proof for Gray versus Binary Encodings. In Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann 1999

[16] D. Whitley, S. Rana and R.B. Heckendorn. Representation Issues in Neighborhood Search and Evolutionary Algorithms .In Genetic Algorithms in Engineering and Computer Science, 1997.

[17] Eckart.Zitzler. Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications. PhD thesis, Swiss Federal Institute of Technology, Suisse, Novembre 1999.

[18] EckartZitzler, Marco Laumanns, and LotharThiele . Improving the Strength Pareto Evolutionary Algorithm .Computer Engineering and Networks Laboratory ,May 2001

[19] Hervé meunier , Algoriuthmes evolutionnaires paralleles pour l'optimisation multi-objectif de résaux de télécomunication mobiles. Thése pour obtenir le grade de Docteur de L U.S.T.L Université des sciences et technàologies de ILLE, 12/06/2002L

[20] J.D. Schaffer .Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, Genetic Algorithms and Their Applications . Proceedings of the First International Conference on Genetic Algorithms, pp. 93-100, 1985.

[21] J. R. Schott. Fault tolerant design using single and multi-criteria genetic algorithms. Master'sthesis, Departement of Aeronautics and Astronautics, Boston,1995.

[22] Jean-Sébastien Lacroix, Stéphane Terrade.Algorithmes Génétiques .Ecole Polytechnique Montréal, 17 Novembre 2004

[23] J. Knowles .Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization .PhD Thesis submitted to the Department of Computer Science, University of Reading, UK, 2001.

[24] kalyanmoy Deb, .Amritpratap, Sameer Agarwal and T.Meyarivan.Rapport.a Fast and eletist multiobjective genetic Algorithm :NSGAII . indian institute of technology Kanpour

[25] Kalyanmoy Deb, Debayan Deb .analyzing Mutation Schemes for Real-Parameter Genetic Algorithms . Department of Mechanical Engineering Dept of Computer Scienceand EngineeringIndian Institute of Technology Kanpur Michigan State ,University Kanpur

[26] K. Deb. Optimization for engineering design : Algorithms and examples. New Delhi Prentice-Hall, 1995.

[27] LAMAMRA Khireddine. Optimisation multi-objectifs par les algorithmes génétiques et application à la commande des systèmes .THESE Pour l'obtention du Diplôme de DOCTORAT ES SCIENCES Spécialité Electronique ,Université Mentouri, Constantine ,01/03/2012

[28] Lamia Benameur .Contribution à l'optimisation complexe par des techniques de swarm intelligence. Mémoire pour l'obtention du titre d'ingénieur en Informatique, Université Mohamed V Agdal Rabat Maroc,2010

[29] L. Barbulescu, J.P. Watson and D. Whitley.Dynamic Representations and Escaping Local Optima : Improving Genetic Algorithms and Local Search. AAAI/IAAI, 2000.

[30] L.J. Eshelman and J.D. Schaffer .Real-coded Genetic Algorithms and Interval Schemata . In Foundations of Genetic Algorithms - 2, pp. 187-202, Morgan Kaufmann ,1993.

[31] M. Ehrgott . A characterization of Lexicographic Max-ordering Solutions, Methods of Multicrite-ria Decision Theory .Proceedings of the 6th Workshop of the DGOR Working Group Multicriteria and Decision Theory, Egelsbach, Häsel-Hohenhausen, pages 193-202, 1997.

[32] MERDJAOUI Brahim. Optimisation multi-objectif par algorithmes génétiques et approche Pareto des paramètre d'usinage sous contraintes des limitations de production .Mémoire de Magister en Génie Mécanique. Université M'Hamed Bougara Boumerdes, 14 Octobre 2006

[33] Miroslav GREGAN . Applications d'algorithmes génétiques (AGs). Cours au sein d'EIDV. Ecole d'ingénieurs du Cancon Du Vaud ,13 Janvier 2005

[34] MouadhYagoubi .Optimisation évolutionnaire multi-objectif parallèle application à la combustion Diesel . T H È S E pour obtenir le titre de Docteur en Sciences de l'Université de Paris Sud XI ,20 septembre 2012

[35] NadiraBenlahrache . Optimisation Multi-Objectif Pour l'Alignement Multiple de Séquences . Mémoire Présenté en vue de l'obtention du diplôme de Magistère en Informatique. Université Mentouri de Constantine ,2007

[36] OlgaRoudenko . Application des Algorithmes Evolutionnaires réseaux Problèmes d'Optimisation Multi-Objectif avec Contraintes .Thése présenter pour obtenir le grade de docteur en mathématiques appliquées de l'école polytechniques ,5 mars 2004

[37] SAADI LEILA .Optimisation Multi-Objectifs par Programmation Génétique . MEMOIRE Présenté Pour l'obtention du diplôme Magister en informatique OPTION INFORMATIQUE INDUSTRIELLE à l'université de BATNA ,08/07 /2007

[38] SalimFettaka. .Application of Multiobjective Optimization in Chemical Engineering Design and Operation. A thesis submitted to the Faculty of Graduate and Post-Doctoral Studies in partial fulfillment of the requirements for the degree of Masters of Applied Science .Department of Chemical and Biological . University of Ottawa ,July 2012

[39] TroudiFatiha . Résolution du problème de l'emploi du temps : Proposition d'un algorithme évolutionnaire multi objectif .Mémoire Présenté pour l'obtention du diplôme de Magister en Informatique. Université Mentouri – Constantine ,2005-2006

[40] T. Vallée. Présentation des algorithmes génétiques et de leurs applications en économie ; IFReDE-E3i . Université Montesquieu Bordeaux IV , Décembre 2003.

[41] Vincent Barichard . Approches Hybrides pour les Problème Multiobjectifs (thèse de Doctorat) , 24 Novembre,Université d'Angers , Page 9, 11,37, 402003

[42] Yann Collette, Patrick Siarry . Optimisation Multiobjectif .Edition Eyrolles, 75240 Paris Cedex 05, France ; Page 1, 1, 2, 17, 18, 19, 21, 22,23, 41, 42, 43, 44, 46. . Septembre 2002 .

Annexe

1. Introduction :

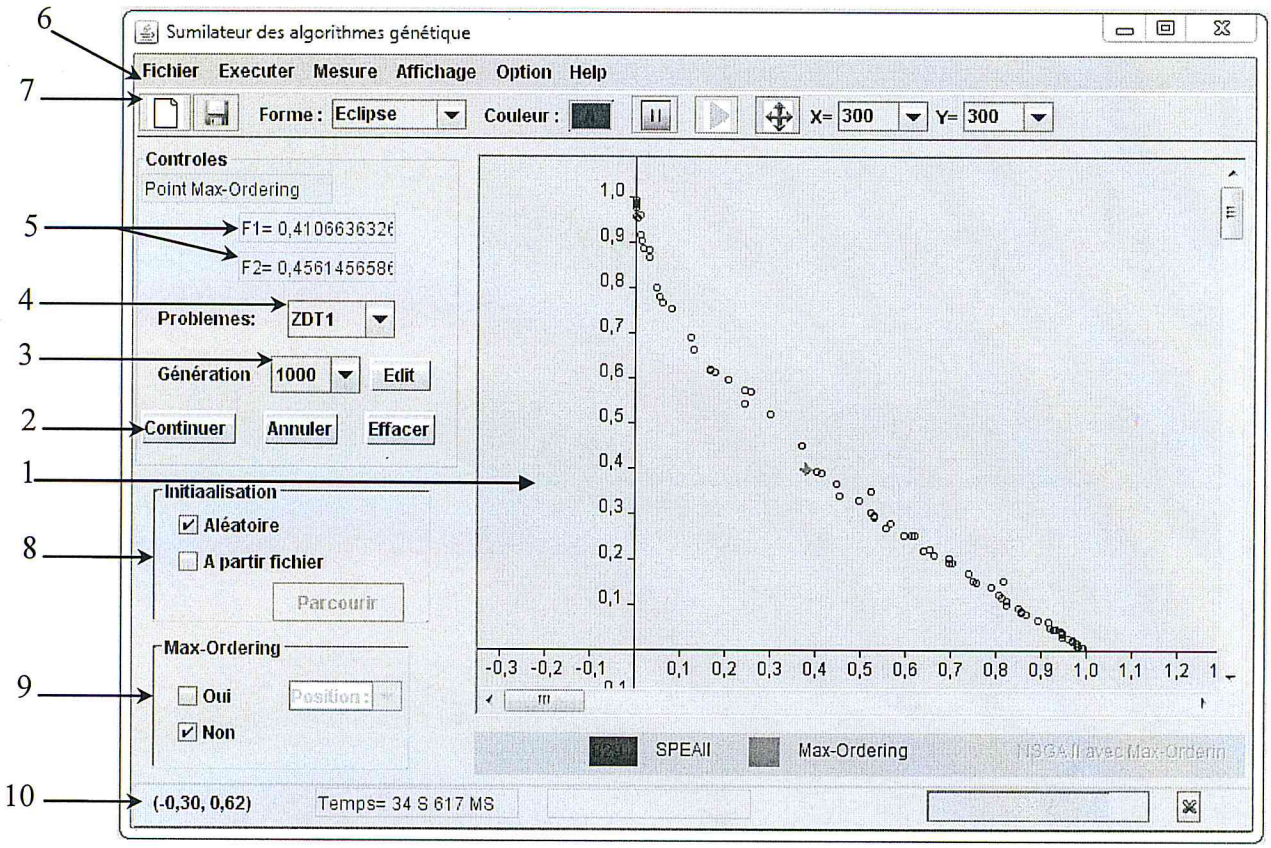
Dans ce chapitre, on va donner une description de l'application (**outil de simulation et comparaison**), cette description aide l'utilisateur à manipuler notre application et de comprendre ces principales fondements.

Le choix de la programmation :

Nous avons choisi le langage JAVA NetBeans IDE 7.0 pour implémenter notre outil de simulation et comparaison, à pour des raisons de portabilité vers d'autres types de machines et avec d'autres systèmes d'exploitation, ainsi que la facilité de réutilisation du code.

4. Présentation de la fenêtre principale de l'interface graphique :

Cette section offre une présentation générale des principaux éléments de l'interface :



1. L'espace de dessin ou fenêtre graphique.
2. Les boutons concernant l'exécution (continuer l'exécution, annuler l'exécution, effacer la fenêtre de dessin).
3. Une composante pour le choix de nombre d'itération.
4. Une composante pour choisir une fonction test.
5. Information sur le point Lexico-Max-Ordering.
6. La barre de menus.
7. La barre d'outils **Standard**.
8. Une composante pour le choix d'initialisation (aléatoire ou à partir d'un fichier).
9. Une composante pour le choix d'initialisation du point Lexico-Max-Ordering.
10. Barre d'état.

4.1. L'espace de dessin ou fenêtre graphique:

L'espace qui affiche l'évolution d'un algorithme.

4.2. Les boutons concernant l'exécution :

Bouton	**Nom de l'outil**	**Fonction**
Continuer	**continuer l'exécution**	Permet de continuer l'exécution d'un algorithme tant qu'on augmente le nombre d'itération.
Annuler	**annuler l'exécution**	Permet d'annuler l'exécution de l'algorithme.
Effacer	**effacer l'espace de dessin**	effacer les courbes qui existent dans l'espace de dessin.

4.3. Choix de nombre d'itération :

Cette composante permet à l'utilisateur de choisir le nombre d'itération d'un algorithme génétique sélectionné.

Remarque :

Quand on clique sur le bouton '**Continuer**', pour continuer l'exécution de l'algorithme, une vérification sera lancée automatiquement comme suit :

Si le nombre d'itération est supérieur à le nombre d'itération de l'algorithme a exécuté alors l'algorithme sera continué l'exécution.

Sinon une boite de dialogue sera affichée pour nous informer.

4.4. Choix d'une fonction test :

Cette composante a pour but est de permettre à l'utilisateur de choisir la fonction test qu'on va la résoudre pendant l'exécution d'un algorithme gétique spécifier.

4.5. Information sur le point Lexico-Max-Ordering :

Dans ce champ il y a deux composantes "**LabelTxt**" qui affichent les coordonnées du point Lexico-Max-Ordering $(F1(\vec{X}), F2(\vec{X}))$.

4.6. La barre de menus :

La barre de menus est composée de six (06) menus de commande : *Fichier*, *Exécuter*, *Mesures*, *Affichage*, *option*, et *Help*.

4.6.1. Fichier:

Ce menu contient quatre (04) opérations : Nouveau, Enregistrer, Enregistrer l'initialisation, Fermer.

a. Nouveau :

Cette opération permet de créer une nouvelle fenêtre de dessin.

b. Enregistrer:

Cette fonction permet de sauvegarder la population du front Pareto trouvée par l'algorithme sous forme d'un fichier texte.

On sauvegarde l'espace de décision avec l'espace de recherche.

c. Enregistrer l'initialisation:

Cette fonction permet de sauvegarder la population initiale qui a initialisée par l'algorithme génétique sous forme d'un fichier texte.

f. Fermer:

Permet de fermer (sortir) la fenêtre de l'application.

4.6.2. Exécuter:

Ce menu est composé de quatre (04) fonctions importantes : *Exécuter*, *Avec Lexico-Max-Ordering, Select un algorithme, NSGA II et SPEA II.*

a. Exécuter:

Permet d'exécuter l'algorithme sélectionné.

b. Avec Lexico-Max-Ordering :

Ce menu '**ChecBox**' à nous permet d'intégrer le point Lexico-Max-Ordering dans la population initiale.

c. Select un algorithme :

Ce menu à nous permet de sélectionner un algorithme génétique.

d. NSGA II et SPEA II :

Ce menu exécute l'algorithme NSGA II et SPEA II au même temps pour la facilité de comparaison entre eux.

4.6.3. Mesure:

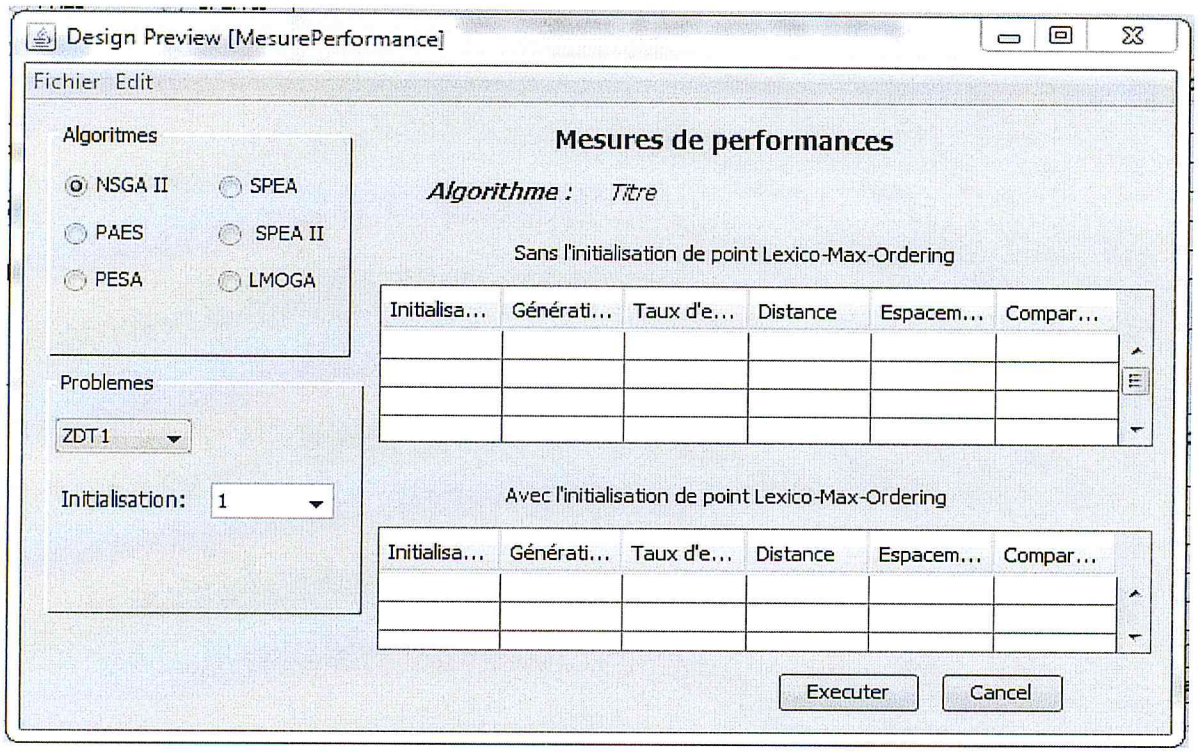
Ce menu contient quatre (04) applications, ces applications sont projetées sur les mesures de performance.

Ces applications sont :

- *Mesure de performance,*
- *Comparaison entre NSGAII et SPEAII,*
- *Calculer et afficher les mesures,*
- *Statistique LMOGA.*

a. Mesure de performance:

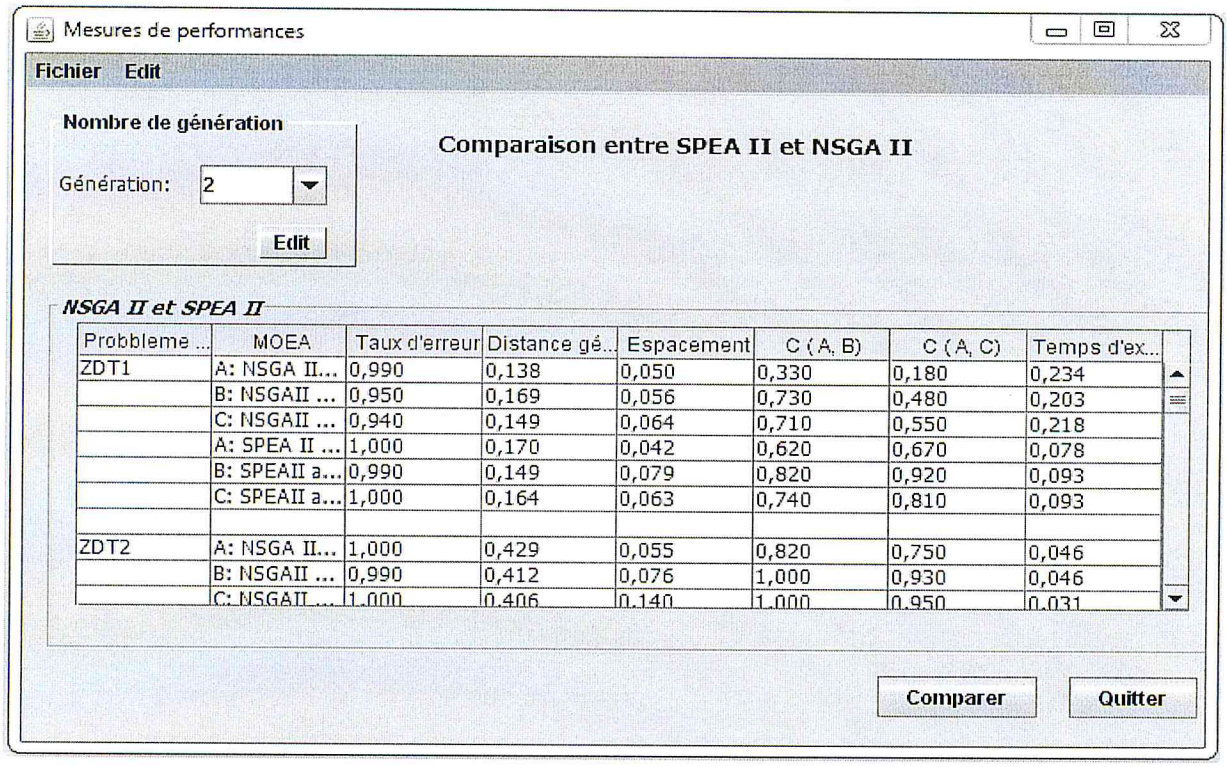
Lorsqu'un algorithme termine son exécution, une population finale sera dessinée dans l'espace de dessin. On peut voir la performance de l'algorithme par la fenêtre qui sera affichée. Elle est montrée dans la figure suivante :



Cette fenêtre permet aussi à l'utilisateur d'exécuter un autre algorithme avec différentes initialisations pour un problème test choisi.

b. Comparaison entre NSGAII et SPEAII :

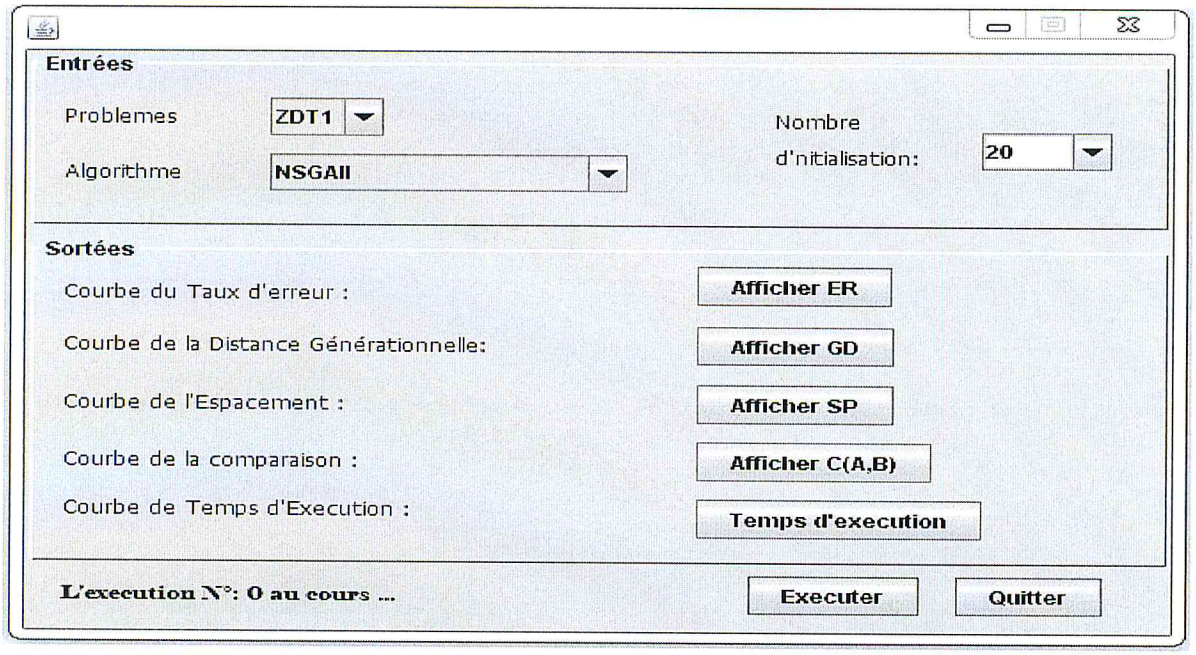
Ce menu permet à l'utilisateur de comparer entre l'algorithme NSGAII et SPEAII avec l'incorporation de point Lexico-Max-Ordering pour toutes les fonctions tests qui existent par la mesure de leurs de performance (Taux d'erreur, Distance générationnelle, Espacement, Comparaison entre deux ensembles). Ces mesures seront affichées dans tableau comme montre la figure suivante :



c. Calculer et afficher les mesures :

Ce menu permet à l'utilisateur de mesurer la performance d'un algorithme pour une fonction test sélectionnée avec plusieurs initialisation et afficher les courbes de la moyenne de ces mesures par rapport le nombre d'itération.

Une fenêtre est affichera après la terminaison de l'exécution de l'algorithme comme montre la figure suivante :

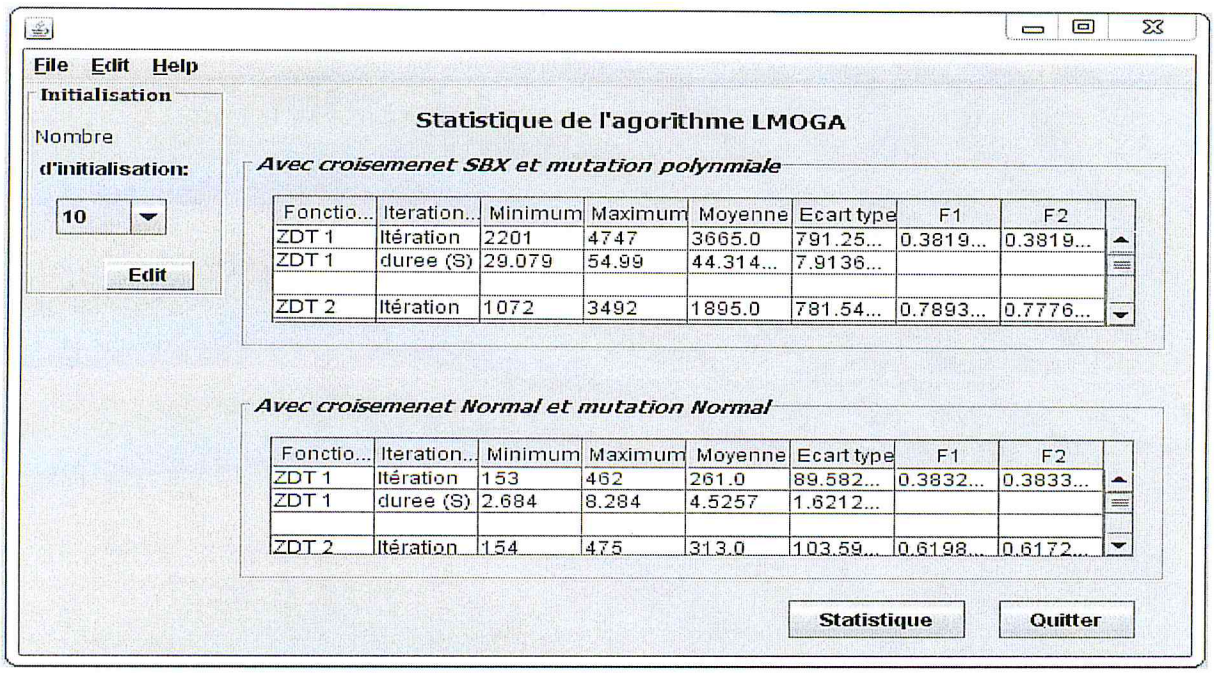


Cette fenêtre contient 5 botons, chaque boton est spécifier pour une mesure particulière.

d. Statistique LMOGA:

Ce menu à nous permet de faire une comparaison entre l'algorithme LMOGA avec les opérateurs de variation DEB el LMOGA avec les opérateurs de variation usuel dans plusieurs initialisation pour toutes les fonctions tests qui existent, par la mesure de la moyenne de nombre d'itération et la moyenne de temps d'exécution

Une fenêtre est affichera après la terminaison de l'exécution de la comparaison comme montre la figure suivante :



4.7. La barre d'outils Standard :

La barre d'outils **Standard** contient plusieurs boutons outils utilisés pour ouvrir, enregistrer et imprimer le graphe. Bon nombre des fonctions de la barre d'outils Standard sont également disponibles dans les menus **Fichier** ou **Edition**.

Bouton	Nom de l'outil	Fonction
	**Nouveau**	Permet de lancer une nouvelle forme de l'application.
	**Enregistrer**	Permet d'enregistrer la courbe qui trouvé par l'algorithme en format fichier texte.
	Forme	Permet de choisir la forme de la courbe.
	**Couleur**	Permet de choisir un couleur de la courbe.
	**Pause/ continuer**	Mettre l'algorithme en pause ou bien continuer l'exécution de l'algorithme.
	**Exécuter**	Exécuter un algorithme sélectionné.
	**Déplacer**	Déplacer le repère.

4.8. Le choix d'initialisation (aléatoire ou à partir d'un fichier):

Dans ce choix il y a deux possibilités pour initialiser un algorithme génétique :

- ✓ ***Initialiser aléatoire :***

  L'application initialise l'algorithme aléatoirement et automatiquement.

- ✓ ***Initialiser à partir d'un fichier :***

L'application initialise l'algorithme à partir d'un fichier texte sélectionné.

Remarque :

Si nous avons choisir la composante 'A partir fichier', une fenêtre des fichiers apparaitre il faut sélectionner un fichier parmi cette liste voulons intégrer ce point.

مكتبة الكلية

4.9. Le choix d'initialisation du point Lexico-Max-Ordering:

Ce choix à nous permet d'intégrer le point Lexico-Max-Ordering dans la population initiale.

Remarque :

Si nous avons choisir la composante 'Oui', il faut déterminer à quelle position nous voulons intégrer ce point.

4.10. La Barre d'état :

Dans la barre d'état de application, on a mis les coordonnées du pointeur du la souris (x, y) et le temps d'exécution de l'algorithme.