Métaheuristiques pour l'optimisation combinatoire multi-objectif : Etat de l'art

El-ghazali Talbi

Abstract

Cet article présente une synthèse des métaheuristiques appliquées à la résolution de problèmes d'optimisation combinatoire multi-objectif, suivant une classification proposée. L'objectif principal de telles méthodes est de générer une variété de solutions Pareto-optimales diversifiées dans l'espace de recherche. Une analyse critique de chaque classe de méthode est présentée. Des réponses à des questions ouvertes comme l'évaluation des performance et la comparaison d'algorithmes d'optimisation multi-objectif, et l'étude des paysages des frontières Pareto sont abordées. Certains axes de recherche future dans ce domaine tel que la conception d'algorithmes parallèles et hybrides sont aussi identifiés.

Keywords

Optimisation combinatoire multi-objectif, Optimalité Pareto, Métaheuristiques, Paysages de recherche, Evaluation de performances, Algorithmes génétiques, Recherche tabou, Recuit simulé.

Ce travail a bénéficié du soutien du CNET au titre du marché No.PE:98-757.33

Laboratoire d'Informatique Fondamentale de Lille, Université des Sciences et Technologies de Lille, Bat. M3 59655 Villeneuve d'Ascq, France. E-mail: talbi@lifl.fr

I. Introduction

De nombreux secteurs de l'industrie (mécanique, chimie, télécommunications, environnement, transport, etc.) sont concernés par des problèmes complexes de grande dimension et multi-critères [1](coûts financiers, qualité de service, etc.) pour lesquels les décisions doivent être prises de façon optimale. Les problèmes d'optimisation rencontrés en pratique sont rarement uni-objectif. Il y a généralement plusieurs critères contradictoires à satisfaire simultanément. L'optimisation multi-critère s'intéresse à la résolution de ce type de problèmes. Elle possède ses racines dans le 19ième siècle dans les travaux en économie de Edgeworth et Pareto [62]. Elle a été utilisée initialement en économie et dans les sciences de management et graduellement dans les sciences pour l'ingénieur.

L'optimisation multi-objectif cherche donc à optimiser plusieurs composantes d'un vecteur fonction coût. Contrairement à l'optimisation uni-objectif, la solution d'un problème multi-objectif (PMO) n'est pas une solution unique, mais un ensemble de solutions, connu comme l'ensemble des solutions Pareto optimales (PO)[2]. Toute solution de cet ensemble est optimale dans le sens qu'aucune amélioration ne peut être faite sur une composante du vecteur sans dégradation d'au moins une autre composante du vecteur. Le premier objectif dans la résolution d'un problème multi-objectif est d'obtenir l'ensemble PO des solutions Pareto optimales (ou d'échantillonner le plus uniformement possible des solutions à partir de l'ensemble PO). La détermination de l'ensemble PO n'est qu'une première phase dans la résolution pratique de PMO, qui nécessite dans un deuxième temps le choix d'une solution à partir de cet ensemble suivant des préférences choisies par le décideur. Le choix d'une solution par rapport à une autre nécessite la connaissance du problème et des nombreux facteurs liés au problème. Ainsi, une solution choisie par un décideur peut ne pas être acceptable pour un autre décideur. Le choix d'une solution peut aussi être remis en cause dans un environnement dynamique. Il est donc utile d'avoir plusieurs alternatives dans le choix d'une solution Pareto optimale.

La difficulté dans la conception d'un solveur de PMO réside dans les faits suivants :

- Il n'y a pas de définition communément admise sur l'optimalité d'une solution comme en optimisatiom uni-objectif. La relation d'ordre entre les solutions du problème n'est que partielle, et le choix final revient au décideur.
- Le nombre de solutions optimales (au sens de Pareto) augmente en fonction de la taille des problèmes et principalement avec le nombre de critères utilisés.
- Pour les PMO non convexes, les solutions Pareto sont localisées dans les frontières et à l'intérieur de l'enveloppe convexe des solutions réalisables.

Une des questions fondamentales dans la résolution de PMO concerne la coopération entre le solveur du problème et le décideur final, qui peut prendre une des trois formes suivantes :

- A priori : les solutions proposées pour résoudre des PMO consistent souvent à combiner les différentes fonctions coût suivant une certaine *fonction d'utilité*, pour obtenir une seule fonction à optimiser. Dans ce cas le décideur est supposé connaître a priori le poids de chaque objectif et ainsi la fonction d'utilité. Ceci revient donc à transformer un PMO en un problème uni-objectif et le résoudre par des méthodes d'optimisation classiques. Cependant, dans la plupart des cas, la fonction d'utilité n'est pas connue a priori du processus d'optimisation, et les différents objectifs ne sont pas commensurables [3]. De plus, l'espace de recherche défini peut ne pas représenter effectivement le problème initial. Si le décideur n'est pas à même d'indiquer a priori le type de compromis qu'il souhaite réaliser entre les critères, il n'est pas pertinent de chercher une et une seule solution efficace réalisant une agrégation entre ces critères.
- A posteriori : le décideur choisit une solution parmi les solutions de l'ensemble PO fourni par le solveur. Cette approche est utilisable dans le cas où la cardinalité de l'ensemble PO est réduite [73]. Dans le cas contraire, pour l'aider à prendre une décision, il convient de lui permettre d'explorer l'ensemble des solutions en fonction de ses préférences, afin qu'il puisse mieux appréhender les arbitrages à opérer entre les critères.
- Interactive : dans ce cas, il y a coopération progressive entre le décideur et le solveur (fig.1). A partir des connaissances acquises pendant la résolution du problème, le décideur définit des préférences. Ces préférences sont prises en compte par le solveur dans la résolution du problème. Ce processus est réitéré pendant plusieurs étapes. A l'issue de l'exploration guidée de l'ensemble PO, le décideur dispose d'une connaissance approfondie pour retenir une solution de l'ensemble PO représentant un compromis acceptable.

Dans cet article, nous nous intéressons aux métaheuristiques permettant d'approximer la frontière Pareto de problèmes d'optimisation combinatoire multi-critères. L'aspect d'aide à la décision pour le choix d'une solution finale parmi les solutions Pareto n'est pas adressé. Cet article généralise les travaux de synthèse réalisés sur l'application des algorithmes évolutionnaires (Coello [9], Fonseca et Fleming [25], et Deb [16]), qui ont reçu un intérêt croissant ces dernières années. Nous présentons un état de l'art généralisé à des métaheuristiques appliquées à la résolution de PMO, suivant une

[1] On utilise indifféremment les termes objectif, attribut et critère.

[2] Cet ensemble est appelé aussi frontière Pareto.

[3] Des objectifs sont non-commensurables si leurs valeurs sont exprimées dans des unités différentes. Par exemple, si on cherche à maximiser le profit et minimiser les dégats écologiques.

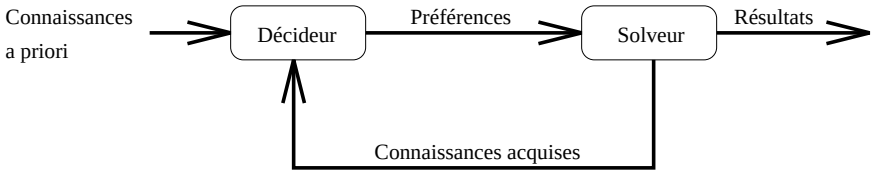


Fig. 1. Approche interactive : Coopération progressive entre le solveur et le décideur.

classification proposée. L'objectif principal de telles méthodes est de générer une variété de solutions Pareto optimales diversifiées dans l'espace de recherche. Une analyse critique de chaque classe de méthode est présentée. Des réponses à des questions ouvertes comme l'évaluation des performances et la comparaison d'algorithmes d'optimisation multi-objectif, et l'étude des paysages des frontières Pareto sont abordées. Certains axes de recherche future dans ce domaine, telle que la conception d'algorithmes hybrides et parallèles, sont aussi identifiés.

L'article est organisé comme suit. Dans la section 2, quelques définitions nécessaires à la compréhension de l'article sont données. Un certain nombre d'exemples de PMO académiques et industriels étudiés dans la littérature sont présentés dans la section 3. La section 4 est consacrée à la classification proposée des méthodes d'optimisation multi-objectif. Dans les section 5, 6 et 7, nous analysons respectivement chacune des méthodes de résolution de PMO : approches à base de transformations d'un PMO en un problème uni-objectif, approches non Pareto à base de populations et les approches Pareto. Les notions de paysages de recherche et d'évaluation de performances sont abordées dans la section 8. Enfin, dans la conclusion, nous présentons quelques perspectives de recherche dans le domaine.

II. Définitions

Un PMO peut être défini de la manière suivante :

$$(PMO) \begin{cases} min\ F(x) = (f_1(x), f_2(x), ..., f_n(x)) \\ \text{s.c. } x \in C \end{cases}$$

où $n \geq 2$ est le nombre de fonctions objectifs, $x = (x_1, ..., x_k)$ est le vecteur représentant les variables de décision, C représente l'ensemble des solutions réalisables associé à des contraintes d'égalité, d'inégalité et des bornes explicites (espace de décision), et $F(x) = (f_1(x), f_2(x), ..., f_n(x))$ est le vecteur des critères à optimiser.

Dans le cadre de l'optimisation multi-objectif, le plus souvent le décideur raisonne plutôt en termes d'évaluation d'une solution sur chaque critère et se place naturellement dans l'espace des critères. L'ensemble $Y = F(C)$ représente les points réalisables dans l'espace des critères (espace objectif) , et $y = (y_1, ..., y_n)$ avec $y_i = f_i(x)$ est un point de l'espace des critères. On impose une relation d'ordre partiel sur cet ensemble de points, appelée *relation de dominance*.

Définition 1: Une solution $y = (y_1, ..., y_n)$ domine une solution $z = (z_1, ..., z_n)$ ssi $\forall i \in [1..n]\ y_i \leq z_i$ et $\exists i \in [1..n] / y_i < z_i$.

Nous avons rarement un vecteur x^* qui est optimum pour tous les objectifs :

$$\forall x \in C, f_i(x^*) \leq f_i(x),\ i = 1, 2, ..., n.$$

Etant donné que cette situation arrive rarement dans les problèmes réels où les critères sont en conflit, d'autres notions ont été établies pour considérer une solution optimale. Le concept généralement utilisé est la notion d'*optimalité Pareto*.

Définition 2: Une solution $x^* \in C$ est Pareto optimale ssi il n'existe pas une solution $x \in C$ tel que $F(x)$ domine $F(x^*)$.

La définition de solution Pareto optimale découle directement de la notion de dominance. Elle signifie qu'il est impossible de trouver une solution qui améliore les performances sur un critère sans que cela entraîne une dégradation des performances sur au moins un autre critère. Les solutions Pareto optimales sont aussi connues sous le nom de solutions *admissibles*, *efficaces*, *non-dominées*, et *non-inférieures*. L'ensemble exact de toutes les solutions Pareto optimales sera noté PO.

Certaines solutions Pareto optimales peuvent être obtenues par la résolution du programme mathématique suivant :

$$(PMO_\lambda) \begin{cases} min\ F(x) = \sum_{i=1}^{n} \lambda_i f_i(x) \\ \text{s.c. } x \in C \end{cases}$$

avec $\lambda_i \geq 0$ pour i=1,..,n, et $\sum_{i=1}^{n} \lambda_i = 1$.

Ces solutions sont dites *supportées*. L'ensemble de ces solutions peut être généré par la résolution de (PMO_λ) pour différentes valeurs du vecteur de poids λ. La complexité de (PMO_λ) est équivalente à celle du problème combinatoire

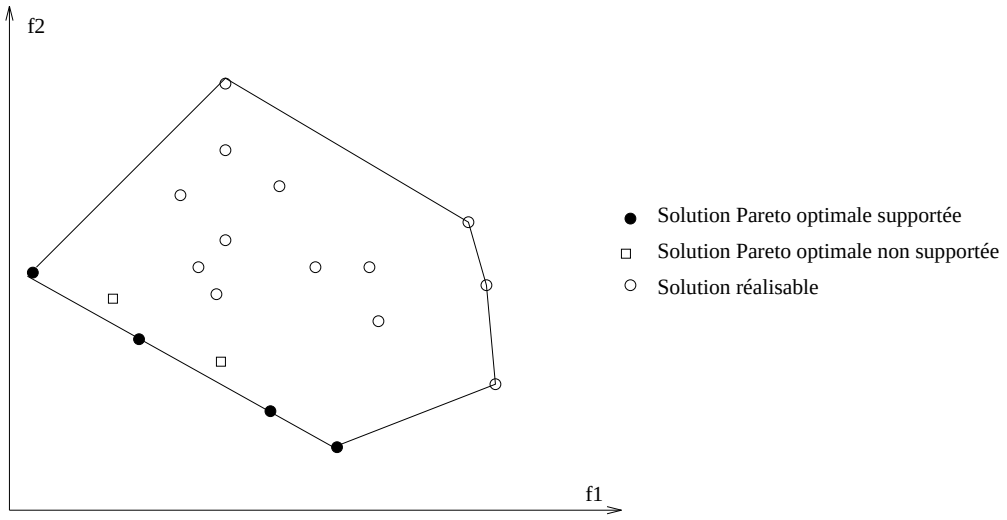


Fig. 2. Solutions supportées et non supportées.

sous-jacent. Dès lors que le problème combinatoire sous-jacent est polynômial (problèmes de cheminement, flot, etc.), l'obtention des solutions supportées est relativement aisée. Néanmoins, il existe en général d'autres solutions qui, bien qu'efficaces, ne peuvent être obtenues par la résolution d'un programme (PMO_λ). En effet, ces solutions, dites *non supportées*, sont dominées par certaines combinaisons convexes de solutions supportées ; il s'agit de points de Y à l'intérieur de l'enveloppe convexe de Y (fig.2).

Il existe d'autres approches permettant de générer ces solutions non supportées (par exemple les approches de programmation par but fondées sur l'utilisation d'une norme de Tchebycheff). Cependant, ces approches modifient la structure du problème combinatoire qui perd ainsi ses éventuelles propriétés remarquables. L'obtention de solutions non supportées est donc en général encore plus difficile.

Supposons que l'optimum pour chaque fonction objectif est connu, la fonction étant optimisée séparément.

Définition 3: Le *vecteur idéal* $y^* = (y_1^*, y_2^*, ..., y_n^*)$ est le vecteur qui optimise chacune des fonctions objectifs f_i, i.e : $y_i^* = min(f_i(x)),\ x \in C$.

Le vecteur idéal est généralement une solution utopique, dans le sens qu'il n'appartient pas à l'espace objectif réalisable. Dans certains cas, le décideur définit un *vecteur de référence* exprimant le but qu'il veut atteindre pour chaque objectif. Ceci généralise la notion de vecteur idéal.

Définition 4: Un *vecteur de référence* $z^* = (z_1^*, z_2^*, ..., z_n^*)$ est un vecteur qui définit le but à atteindre pour chaque objectif f_i.

Enfin, nous allons présenter à l'aide d'un exemple illustratif la notion d'optimum Pareto local. Pour cela, commençons par définir la notion de voisinage qui est utilisée dans les métaheuristiques de recherche locale comme la recherche tabou.

Définition 5: Un voisinage N est une fonction $N : C \longrightarrow P(C)$, qui associe pour chaque $x \in C$ un sous-ensemble $N(x)$ de C des voisins de x.

Ce voisinage est défini par un opérateur de recherche locale. Dans la figure 3, 10 solutions sont représentées dans un espace objectif bi-dimensionnel. Les segments connectant les solutions représentent la structure du voisinage.

Définition 6: Une solution x est localement Pareto optimale ssi $\forall w \in N(x)$, w ne domine pas x.

Dans la figure 3, les solutions Pareto optimales sont associées aux points 1, 8 et 9, et les solutions 4 et 10 sont localement Pareto optimales.

III. Problèmes d'optimisation multi-objectif

Dans les 30 dernières années, la plupart des travaux ont porté sur la programmation multi-objectif linéaire en variables continues. Les raisons principales de cet intérêt sont d'une part le développement de la programmation linéaire uni-objectif en recherche opérationnelle, et la facilité relative de traiter de tels problèmes, et d'autre part l'abondance des cas pratiques qui peuvent être formulés sous forme linéaire. Ainsi, un certain nombre de logiciels ont vu le jour depuis le développement de la méthode du simplexe multi-objectif [101][80].

Nous nous intéressons dans cet article, principalement, aux problèmes d'optimisation combinatoire multi-objectif pour lesquels relativement peu de travaux ont été réalisés. Dans cette section, nous présentons quelques exemples aussi bien académiques qu'industriels.

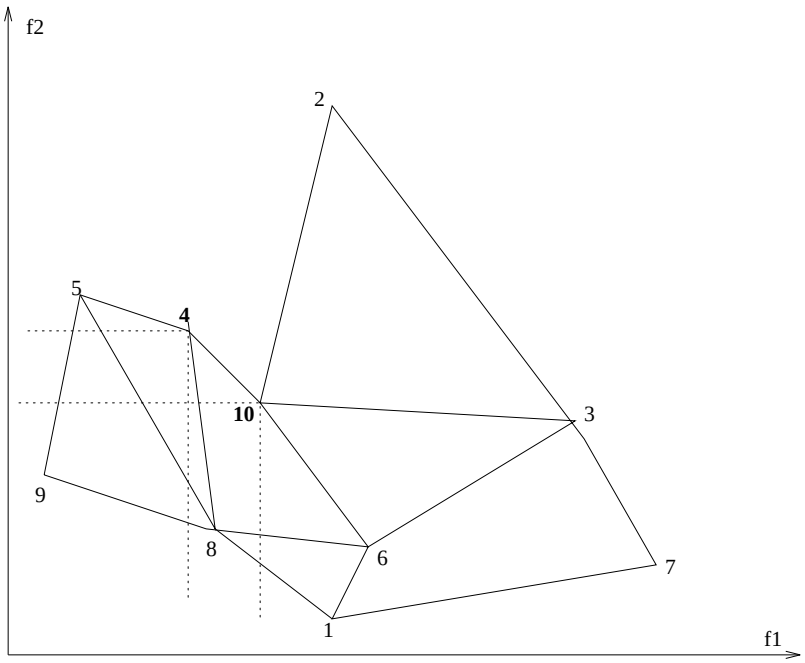


Fig. 3. Optimum Pareto local.

A. Applications académiques

La plupart des benchmarks utilisés dans la comparaison d'algorithmes d'optimisation multi-objectif ont été réalisés sur des problèmes académiques. Les problèmes les plus étudiés sont :

- Optimisation de fonctions continues : dans la communauté "Algorithmes Génétiques", les fonctions de Schaffer sont souvent utilisées pour évaluer et comparer les performances des algorithmes. La fonction bi-critère f_2 est la plus utilisée :

$$\begin{cases} f_{21}(x) = x^2 \\ f_{22}(x) = (x-1)^2 \\ \text{s.c. } x \in [-10, 10] \end{cases}$$

  Pour ce problème, la frontière Pareto est continue et se trouve dans l'intervalle [0..2].
- Optimisation combinatoire : plusieurs problèmes classiques d'optimisation combinatoire ont été étudiés dans une version multi-objectif [13] : sac à dos [1], ordonnancement [59][73], plus court chemin dans un graphe [96], arbre recouvrant (minimum spanning tree) [102], affectation [90], voyageur de commerce [76], routage de véhicules [63], etc.

Prenons comme exemple illustratif le problème du sac à dos multi-objectif qui peut être modélisé de la manière suivante [1] :

$$\begin{cases} Max(f_i(x)) = \sum_{j=1}^{m} c_j^i x_j \\ (i = 1, ..., n), x \in C \\ C = \{x / \sum_{j=1}^{m} w_j x_j < w \\ x_j \in \{0, 1\}, \forall j = 1, ..., m\} \end{cases}$$

où

$$x_j = \begin{cases} 1 & \text{si l'élément } j \text{ est dans le sac} \\ 0 & \text{sinon} \end{cases}$$

w_j le poids (volume) de l'élément j, et c_j^i l'utilité de l'élément j par rapport au critère i.

B. Applications industrielles

La plupart des travaux portant sur l'optimisation combinatoire multi-objectif concernent des applications industrielles. Plusieurs problèmes ont été traités dans différents domaines d'application :

- Design de systèmes dans les sciences pour l'ingénieur (mécanique, aéronautique, chimie, etc.) : ailes d'avions [60], moteurs d'automobiles [29].
- Ordonnancement et affectation : ordonnancement en productique [77][86], localisation d'usines, planification de trajectoires de robots mobiles [28], etc.
- Transport : gestion de containers [87], design de réseaux de transport [27], tracé autoroutier, etc.

- Environnement : gestion de la qualité de l'air [51], distribution de l'eau [36], etc.
- Télécommunications : design d'antennes [94], design de réseaux cellulaires [57], design de constellation de satellites [21], affectation de fréquences [15], etc.

Une liste plus complète d'exemples d'application peut être trouvée dans [93].

IV. Classification des méthodes

Dans la littérature, une attention particulière a porté sur les problèmes à deux critères en utilisant les *méthodes exactes* tels que le *"branch and bound"* [75][91][95][73], *l'algorithme A** [81][56], et la *programmation dynamique* [97][6]. Ces méthodes sont efficaces pour des problèmes de petites tailles. Pour des problèmes à plus de deux critères ou de grandes tailles, il n'existe pas de procédures exactes efficaces, étant donné les difficultés simultanées de la complexité NP-difficile, et le cadre multi-critère des problèmes.

Des méthodes heuristiques sont nécessaires pour résoudre les problèmes de grandes tailles et/ou les problèmes avec un nombre de critères supérieur à deux. Elles ne garantissent pas de trouver de manière exacte l'ensemble PO, mais une approximation de cet ensemble notée PO^*. Les méthodes heuristiques peuvent être divisées en deux classes : d'une part les algorithmes spécifiques à un problème donné qui utilisent des connaissances du domaine [30], et d'autre part les algorithmes généraux applicables à une grande variété de PMO. Notre intérêt dans cet article porte sur la deuxième classe d'algorithmes, les *métaheuristiques*. Plusieurs adaptations de *métaheuristiques* ont été proposées dans la littérature pour la résolution de PMO et la détermination des solutions Pareto : le *recuit simulé* [89], la *recherche tabou* [31] et les *algorithmes évolutionnaires* (algorithmes génétiques [79][23], stratégies évolutionistes [47]).

Les approches utilisées pour la résolution de PMO peuvent être classées en trois catégories (fig.4) :

- Approches basées sur la transformation du problème en un problème uni-objectif : Cette classe d'approches comprend par exemple les méthodes basées sur l'agrégation qui combinent les différentes fonctions coût f_i du problème en une seule fonction objectif F. Ces approches nécessitent pour le décideur d'avoir une bonne connaissance de son problème.
- Approches non-Pareto : Les approches non-Pareto ne transforment pas le PMO en un problème uni-objectif. Elles utilisent des opérateurs de recherche qui traitent séparément les différents objectifs.
- Approches Pareto : Les approches Pareto utilisent directement la notion d'optimalité Pareto dans leur processus de recherche. Le processus de sélection des solutions générées est basé sur la notion de non-dominance.

Dans les sections suivantes, nous présentons respectivement les trois classes de méthodes.

V. Méthodes à base de transformation du problème vers l'uni-objectif

Dans la résolution de PMO, plusieurs méthodes traditionnelles transforment le PMO en un problème uni-objectif. Parmi ces méthodes on trouve les méthodes d'agrégation, les méthodes ϵ-contrainte, et les méthodes de programmation par but (goal programming).

A. Méthode d'agrégation

C'est l'une des premières méthodes utilisée pour la génération de solutions Pareto optimales. Elle consiste à transformer le problème (PMO) en un problème (PMO_λ) qui revient à combiner les différentes fonctions coût f_i du problème en une seule fonction objectif F généralement de façon linéaire [42][79]:

$$F(x) = \sum_{i=1}^{n} \lambda_i f_i(x)$$

où les poids $\lambda_i \in [0..1]$ et $\sum_{i=1}^{n} \lambda_i = 1$. Différents poids fournissent différentes solutions supportées. La même solution peut être générée en utilisant des poids différents.

La figure 5 illustre le fonctionnement de la méthode d'agrégation linéaire. Fixer un vecteur poids revient à trouver un hyper-plan dans l'espace objectif (une ligne pour un problème bi-critère) avec une orientation fixée. La solution Pareto optimale est le point où l'hyper-plan possède une tangente commune avec l'espace réalisable (point x dans la figure).

Les résultats obtenus dans la résolution du problème (PMO_λ) dépendent fortement des paramètres choisis pour le vecteur de poids λ. Les poids λ_i doivent aussi être choisis en fonction des préférences associées aux objectifs, ce qui est une tâche délicate. Ainsi, une approche généralement utilisée consiste à résoudre le problème (PMO_λ) avec différentes valeurs de λ.

Si les différents objectifs sont non-commensurables, on peut transformer l'équation précédente sous la forme :

$$F(x) = \sum_{i=1}^{n} c_i \lambda_i f_i(x)$$

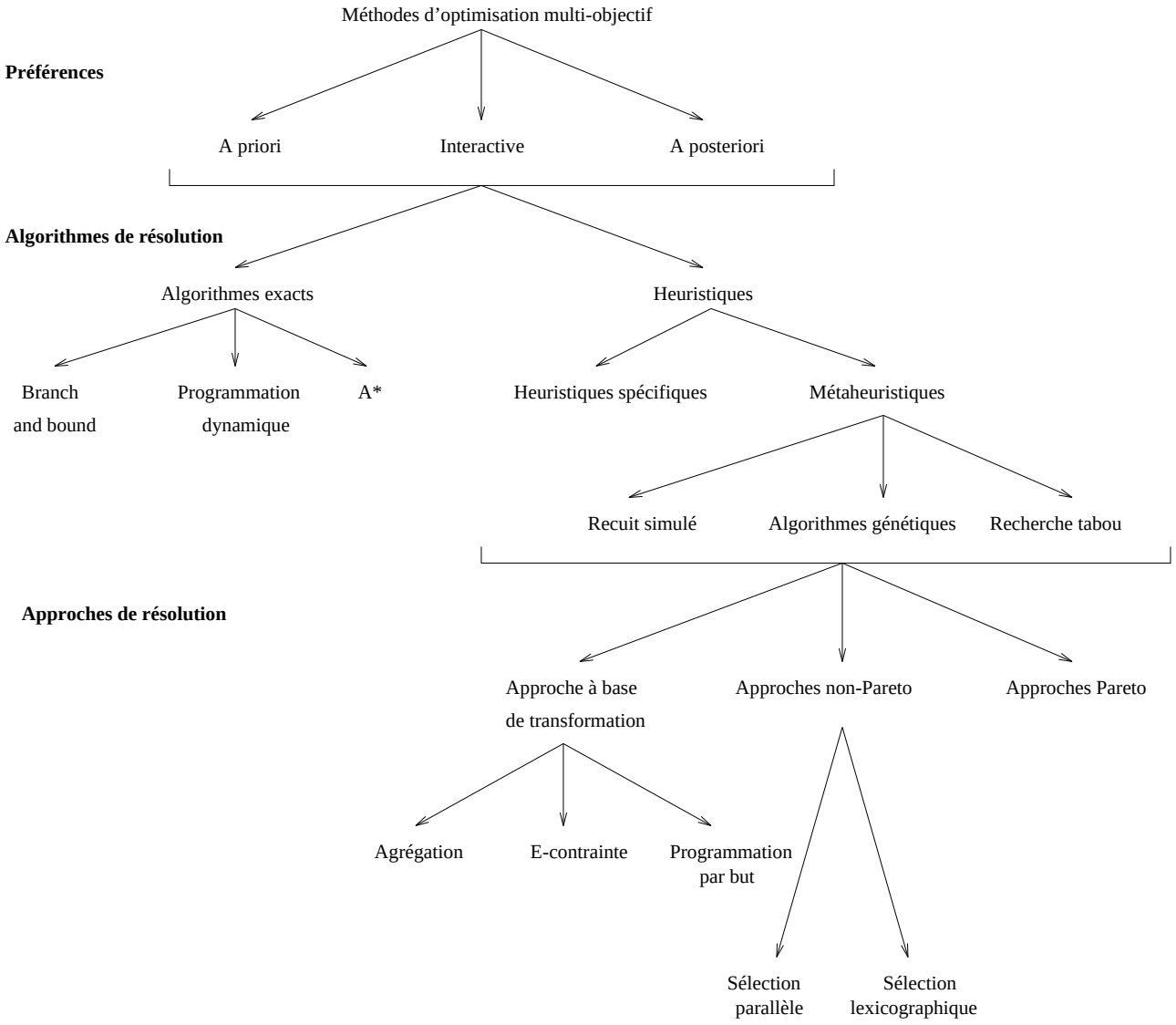
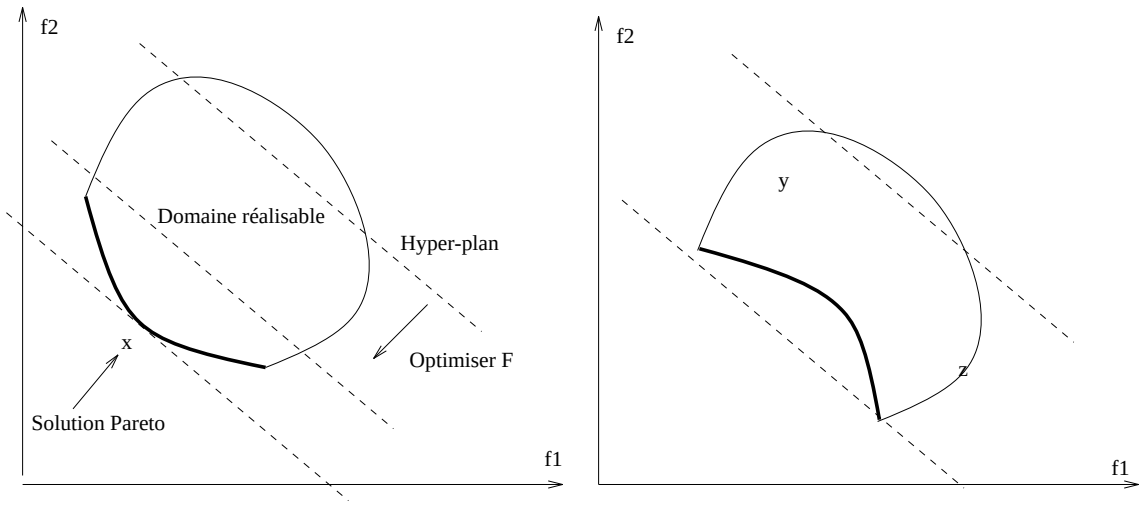


Fig. 4. Classification des méthodes d'optimisation multi-objectif.



(a) Une frontière Pareto convexe

(a) Une frontière Pareto non convexe

Fig. 5. La méthode d'agrégation.

où c_i sont des constantes qui mettent à la même échelle les différents objectifs. Les constantes c_i sont généralement initialisées à $\frac{1}{f_i(x^*)}$, où $f_i(x^*)$ est la solution optimale associée à la fonction objectif f_i. Dans ce cas, le vecteur est normalisé par rapport au vecteur idéal.

L'avantage de ces approches est la production d'une seule solution et ne nécessitent donc pas d'interaction avec le décideur. Cependant, la solution trouvée peut ne pas être acceptable. L'espace de recherche est réduit de façon prématurée avant que des informations suffisantes soient disponibles. L'autre problème avec cette approche est la détermination des poids, sans avoir de connaissances sur le problème traité. Plusieurs stratégies "aveugles" peuvent être utilisées pour générer les poids. Dans [43], les poids sont générés de façon aléatoire :

$$w_i = \frac{random_i}{random_1 + ... + random_n},\ i = 1, 2, ..., n$$

où les variables $random_i$ sont des entiers positifs.

L'exécution multiple de l'algorithme avec des poids différents permet d'obtenir plusieurs solutions efficaces. Cependant, elle ne génère que des solutions supportées, et non pas les solutions dans les portions concaves de l'espace de recherche (solutions non supportées). Dans la figure 5b, où le problème possède une frontière Pareto non convexe, seules les solutions y et z peuvent être générées. Toutes les autres solutions entre les points x et y ne peuvent être trouvées.

Néanmoins, cette approche a largement été utilisée dans la littérature à l'aide de différentes métaheuristiques :

- **Algorithmes génétiques** : les algorithmes génétiques ont été utilisés pour résoudre plusieurs PMO transformés en un problème uni-objectif [9] : ordonnancement [83], planification de robots [44], génération de structures chimiques [46], placement [49], transport [100].
  Dans [35], l'algorithme proposé intègre dans le codage de chaque individu les poids de chaque objectif, en plus de la représentation d'une solution du problème dans le codage d'un individu. L'avantage de cette approche est qu'elle encourage la diversité des poids utilisés dans la population à travers une fonction de partage (voir section VII-B.3). Le but de cette approche est de générer en parallèle un ensemble de solutions Pareto optimales correspondant à différents poids. La fonction objectif utilisée possède la forme suivante :

  $$F = \sum_{i=1}^{n} \lambda_i \frac{f_i}{f_i^*}$$

  où f_i^* est le paramètre de normalisation de l'objectif f_i, et λ_i sont les poids associés à chaque objectif f_i.
- **Recuit simulé** : l'algorithme du recuit simulé a été utilisé pour le voyageur de commerce multi-objectif [76], pour le design de réseaux de transport [27], et pour le problème du sac à dos multi-objectif [92], où la fonction d'acceptation d'une solution voisine est de la forme :

  $$P_{xy}(T) = min(1, e^{\frac{\sum_{i=1}^{n} \lambda_i (f_i(x) - f_i(y))}{T}})$$

  où x est la solution courante, y est la solution voisine générée, et T est la température courante. Plusieurs exécutions sont réalisées pour différents poids.
- **Recherche tabou** : la recherche tabou a été appliquée pour résoudre un problème d'affectation de fréquences [15]. L'algorithme est exécuté avec un seul jeu de poids qui est fixé suivant les priorités des objectifs.
- **Métaheuristiques hybrides** : récemment, des métaheuristiques hybrides ont été proposées. Dans [88], l'algorithme utilisé est le recuit simulé, mais les solutions initiales sont générées par un algorithme glouton. Dans [43], un algorithme hybridant un AG et une recherche locale est utilisé. Dans l'AG, chaque sélection d'une paire d'individus se fait avec des poids différents. Une recherche locale est effectuée à partir de l'individu produit par l'AG. La direction de la recherche est déterminée par les poids utilisés dans la sélection de l'individu.

B. Méthode ϵ-contrainte

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

$$(PMO_k(\epsilon)) \begin{cases} min\ f_k(x) \\ x \in C \\ \text{s.c. } f_j(x) \leq \epsilon_j,\ j = 1, .., n,\ j \neq k \end{cases}$$

où $\epsilon = (\epsilon_1, ..., \epsilon_{k-1}, \epsilon_{k+1}, ..., \epsilon_n)$.

Ainsi, un problème uni-objectif (objectif f_k) sujet à des contraintes sur les autres objectifs est résolu. Différentes valeurs de ϵ_i peuvent être données pour pouvoir générer différentes solutions Pareto optimales. La connaissance a priori

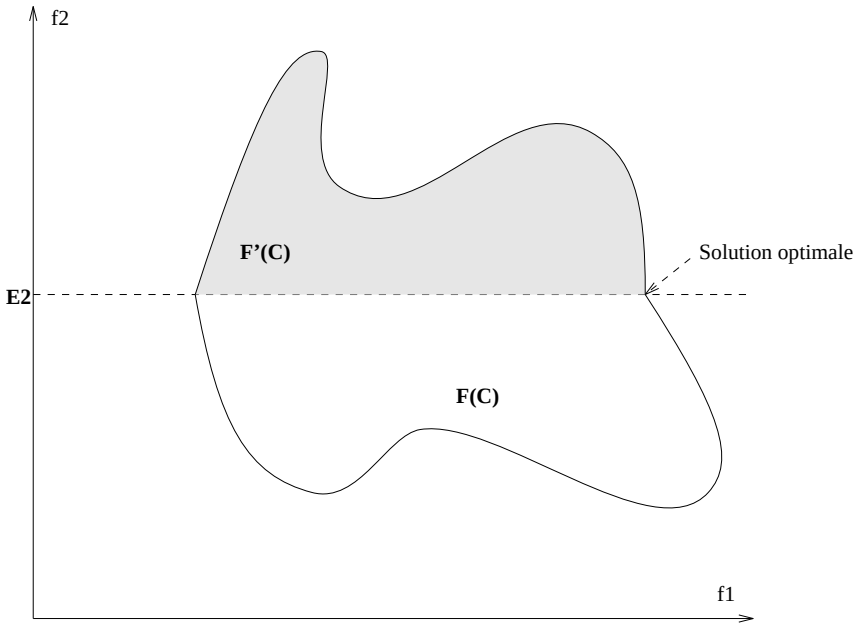


Fig. 6. La méthode ϵ-contrainte.

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

$$\epsilon_i \geq f_i(x^*),\ i = 1, 2, k-1, k+1, n$$

La figure 6 illustre la méthode ϵ-contrainte pour un problème de maximisation bi-objectif, où $F(C)$ est l'espace objectif du problème original, restreint à $F^{'}(C)$ par la transformation du problème, en maximisant la fonction f_1 et en rajoutant la contrainte $f_2(x) \geq \epsilon_2$. Cette méthode peut trouver des solutions non supportées.

Un ordre lexicographique peut être établi entre les objectifs. Par exemple, l'ensemble PO^* peut être obtenu en minimisant les objectifs, commençant par le plus prioritaire et suivant l'ordre décroissant des priorités. Supposons que les indices des fonctions objectifs désignent aussi la priorité ; la fonction f_1 est donc la plus prioritaire. Le premier problème résolu sera formulé de la manière suivante :

$$Min\ f_1(x),\ \text{s.t. } x \in C.$$

Les contraintes n'ont pas été prises en compte. Soit x_1^* la solution optimale trouvée. Le deuxième problème traité serait :

$$Min\ f_2(x),\ \text{s.t. } x \in C,\ \text{avec } f_1(x) = f_1(x_1^*)$$

Une contrainte d'égalité est associée aux fonctions déjà optimisées. Ce processus est itéré jusqu'au traitement de la fonction f_n. La solution finale du problème sera x_n^*. La figure 7 illustre le déroulement de cette procédure dans le cas de problèmes bi-objectifs.

L'approche ϵ-contrainte a été expérimentée en utilisant différentes métaheuristiques :

- **Algorithmes génétiques** : plusieurs travaux ont utilisé les AGs pour résoudre cette classe de problèmes [94]. Dans [70], un algorithme génétique est exécuté plusieurs fois avec différentes valeurs du vecteur ϵ pour générer différentes solutions Pareto optimales.
  Loughlin et Ranjithan ont proposé un AG où les valeurs du vecteur ϵ sont variables suivant les individus [51][50]. Considérons un problème bi-objectif, les seuils associés respectivement à l'individu 1 et l'individu k vont être initialisés au seuil minimal ϵ_{min} et au seuil maximal ϵ_{max}. Le seuil associé à l'individu i de la population aura la forme suivante :
  $$\epsilon_i = \epsilon_{min} + \frac{(i-1).(\epsilon_{max} - \epsilon_{min})}{(k-1)}$$
  où k est la taille de la population.
- **Recherche tabou** : dans [38], la recherche tabou est utilisée pour résoudre une suite de sous-problèmes à un seul objectif sous plusieurs contraintes, dans lesquels chaque objectif est considéré tour à tour, en suivant leur ordre

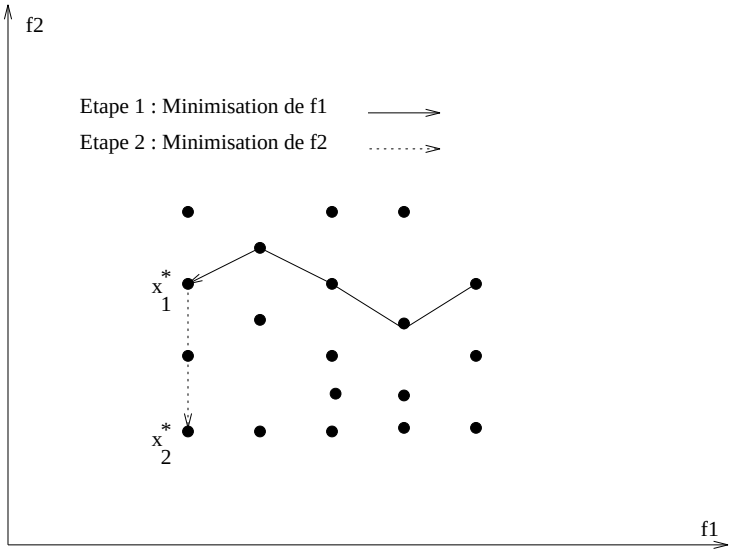


Fig. 7. Ordre lexicographique dans le cas bi-critères.

d'importance relative (ordre lexicographique). En fait, la recherche tabou va chercher une bonne solution pour la fonction semblant la plus importante. Ensuite, la deuxième fonction dans l'ordre relatif d'importance va être optimisée en ayant comme contrainte supplémentaire de ne pas trop détériorer la valeur obtenue pour la première fonction objectif (application d'un seuil). Ce procédé est itéré pour les autres fonctions. Les auteurs testent leur approche pour des problèmes bi-critères. Le problème (PMO_q) résolu à l'étape $q = 1, .., n$ de ce schéma est :

$$(PMO_q) \begin{cases} f_q^* = min\ f_q(x) \\ \text{s.c. } f_r(x) \leq f_r^{'},\ r = 1, .., q-1 \\ x \in C \end{cases}$$

où $f_r^{'}$ est le seuil sur la valeur de l'objectif f_r, calculée à partir de l'optimum f_r^* du problème PMO_r plus une détérioration maximale acceptée pour l'objectif f_r. L'importance relative de chaque objectif peut être contrôlée en transformant la valeur du seuil.

- **Métaheuristiques hybrides** : une méthode combinant les AGs avec la recherche locale (méthode du gradient) a été proposée dans [65].

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

Il existe une relation entre le modèle basé sur l'agrégation et le modèle basé sur la méthode ϵ-contrainte.
Théorème 1: [Chankong et Haimes 1983] Supposons que C et f sont convexes. Si, pour un certain k, x^* est solution de $PMO_k(\epsilon)$, il existe alors λ tel que x^* est solution de PMO_λ, et inversement.

Plusieurs hybridations de ces deux modèles ont été proposées. Ci-dessous un exemple de modèle combinant le modèle PMO_λ et le modèle $PMO_k(\epsilon)$:

$$(PMO(\lambda, \epsilon)) \begin{cases} min\ F(x) = \sum_{i=1}^{n} \lambda_i f_i(x) \\ \text{s.c. } x \in C \\ f_j(x) \leq \epsilon_j,\ j = 1, .., n \end{cases}$$

C. Programmation par but

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

$$(PMO(\lambda, z)) \begin{cases} min(\sum_{j=1}^{n} \lambda_j |f_j(x) - z_j|^p)^{\frac{1}{p}} \\ \text{s.c. } x \in C \end{cases}$$

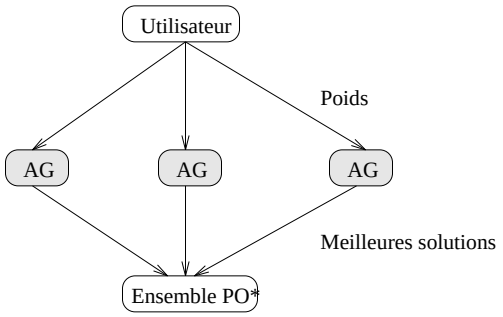


Fig. 8. Exécution parallèle associée à plusieurs agrégations.

où $1 \leq p \leq \infty$, et z est le vecteur de référence (but) ou le vecteur idéal. La norme utilisée est la métrique de Tchebycheff (L_p-métrique). Généralement p est égal à 2 ; dans ce cas on a une métrique euclidienne. Si $p = \infty$, l'équation revient à une fonction min-max. Une sélection arbitraire du vecteur de référence peut ne pas être désirable, puisqu'un mauvais vecteur de référence peut aboutir à une solution qui n'est pas Pareto optimale.

Différentes métaheuristiques ont été utilisées pour résoudre cette classe de problèmes :

- **Algorithmes génétiques** : le problème ainsi formulé a été traité par les algorithmes génétiques [98][72].
  Dans [11][10], une fonction min-max comparant les déviations relatives par rapport à des minimas atteignables (vecteur idéal) est utilisée. Considérons la fonction objectif f_i où la déviation peut être calculée de la façon suivante :

$$z_i^{'}(x) = \frac{|f_i(x) - f_i^0|}{|f_i^0|}$$

  ou

$$z_i^{''}(x) = \frac{|f_i(x) - f_i^0|}{|f_i(x)|}$$

  On suppose que $\forall x, f_i(x) \neq 0$. La fonction à optimiser est alors :

$$F(x) = \sum_{i=1}^{n} \lambda_i |\frac{f_i(x) - f_i^0}{\alpha_i}|$$

  avec $\alpha_i = f_i^0$ ou $f_i(x)$, n est le nombre d'objectifs, et λ_i les poids associés à la fonction d'agrégation.
  L'utilisateur fournit plusieurs jeux de poids λ_L. Un AG par jeu de poids est exécuté (fig.8). Chaque AG résoud le problème d'optimisation avec l'agrégation associée. A la fin de l'exécution de tous les AGs, l'ensemble des solutions Pareto optimales PO^* est produit à partir de la meilleure solution trouvée par chacun des AGs.
  Une procédure de sélection basée sur le paradigme de la *logique flou* (fuzzy logic) a été proposée dans [66][67]. Le calcul de la fonction d'utilité F est basé sur des règles floues :

$$F = \frac{1}{n} \sum_{i=1}^{n} f^{'}(f_i)$$

  où $f^{'}$ représente l'ensemble des règles floues. Il peut être défini comme suit :

$$\begin{cases} \text{Si } f_i \leq (O_i - E_i) \text{ Alors } f^{'}(f_i) = (\frac{S_{min}}{f_{imin} - (O_i - E_i)})(f_i - (O_i - E_i)) \\ \text{Si } (O_i - E_i) \leq f_i \leq (O_i + E_i) \text{ Alors } f^{'}(f_i) = 0 \\ \text{Si } f_i \geq (O_i + E_i) \text{ Alors } f^{'}(f_i) = (\frac{-S_{max}}{(O_i + E_i) - f_{imax}})(f_i - (O_i + E_i)) \end{cases}$$

  où O_i est le but associé à chaque objectif, E_i est l'erreur ou l'incertain accepté pour O_i, $S_{min(max)}$ est le facteur d'échelle pour les valeurs inférieures (supérieures) à la valeur acceptée, $f_{min(max)}$ est la borne inférieure (supérieure) pour chaque objectif. La figure 9 illustre ces règles. Cette approche permet le calcul incertain de chaque objectif avec un seuil de tolérance égal à O_i.
- **Recuit simulé** : dans [76], le recuit simulé a été utilisé pour résoudre le problème du voyageur de commerce bi-objectif avec la fonction d'acceptation suivante :

$$P_{xy}(T) = min(1, e^{\frac{max_i(\lambda_i(f_i(x) - z_i)) - max_i(\lambda_i(f_i(y) - z_i))}{T}})$$

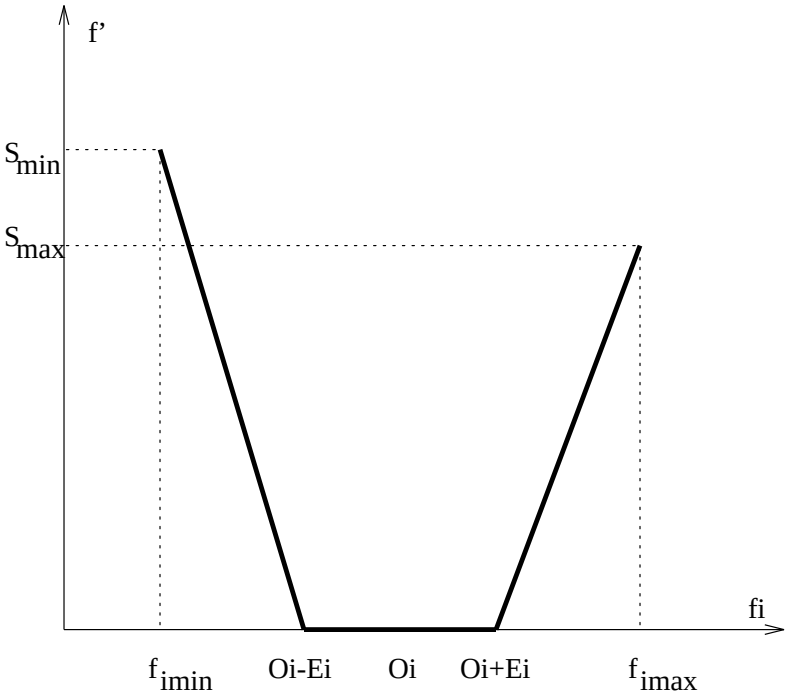


Fig. 9. Règle floue appliquée aux fonctions objectifs.

où x est la solution courante, y est la solution générée dans le voisinage de x, la norme utilisée est la norme L_∞ de Tchebytcheff, et z_i est la valeur de référence. Les poids λ_1 et λ_2 sont initialisés à 1, et à chaque itération un nombre aléatoire généré dans l'intervalle $[-0.05, +0.05]$ permet de mettre à jour les deux poids. Ceci permet une variation lente des poids durant la recherche, et ainsi la détermination de plusieurs solutions Pareto optimales.

- **Recherche tabou** : dans [31], la recherche tabou utilisée est classique, si ce n'est la détermination du choix du meilleur voisin. La solution sélectionnée est celle qui représente le meilleur compromis parmi toutes les solutions du voisinage qui ne sont pas tabous. Pour déterminer cette solution, une distance est introduite de telle sorte que le voisin accepté "contente" toutes les fonctions objectifs. La fonction coût utilisée pour sélectionner le meilleur voisin est une norme pondérée (norme L_p de Tchebycheff) par rapport au vecteur idéal. Tout au long du processus, on garde en mémoire un nombre M de solutions. Toute solution acceptée lors de la recherche tabou est comparée aux M solutions mémorisées. Si une ou plusieurs solutions sont dominées, elles sont éliminées de cet ensemble et remplacées par de nouvelles (pour le moment non dominées). Finalement, une solution est proposée, celle qui représente le meilleur compromis parmi les M solutions sauvegardées (en utilisant la notion de distance précédemment mentionnée).

Les approches basées sur la programmation par but trouvent une solution non dominée si le but est choisi dans le domaine réalisable. Cependant, le décideur est chargé de choisir le vecteur but et les poids associés à chaque objectif, ce qui est une tâche difficile sans connaître la structure de l'espace de recherche. Si le domaine réalisable n'est pas facile à approcher, la méthode peut être inefficace.

D'autres approches basées sur la spécification des préférences par l'intermédiaire d'un but à atteindre ont été proposées dans la littérature comme par exemple le "Goal attainment".

Dans cette approche, un vecteur de poids $\lambda_1, \lambda_2, ..., \lambda_n$ et les buts désirés $b_1, b_2, ... b_n$ pour tous les objectifs doivent être choisis par le décideur. Pour trouver la meilleure solution de compromis x^*, le problème suivant est résolu :

$$\begin{cases} min\ \alpha \\ \text{s.c. } x \in C \\ f_i(x) \leq b_i + \alpha\lambda_i,\ i = 1, .., n \\ \sum_{i=1}^{n} \lambda_i = 1 \end{cases}$$

où α est une variable scalaire. Si un poids λ_i est nul, la limite maximale de l'objectif $f_i(x)$ est donc b_i.

Il a été montré dans [8] que l'ensemble PO peut être généré en variant les poids λ_i, même pour les problèmes non convexes. L'approche est illustrée par la figure 10. Le vecteur b représente le but déterminé par le décideur, qui choisit aussi la direction du vecteur λ. La direction du vecteur $b + \alpha\lambda$ peut être déterminée, et le problème revient à trouver

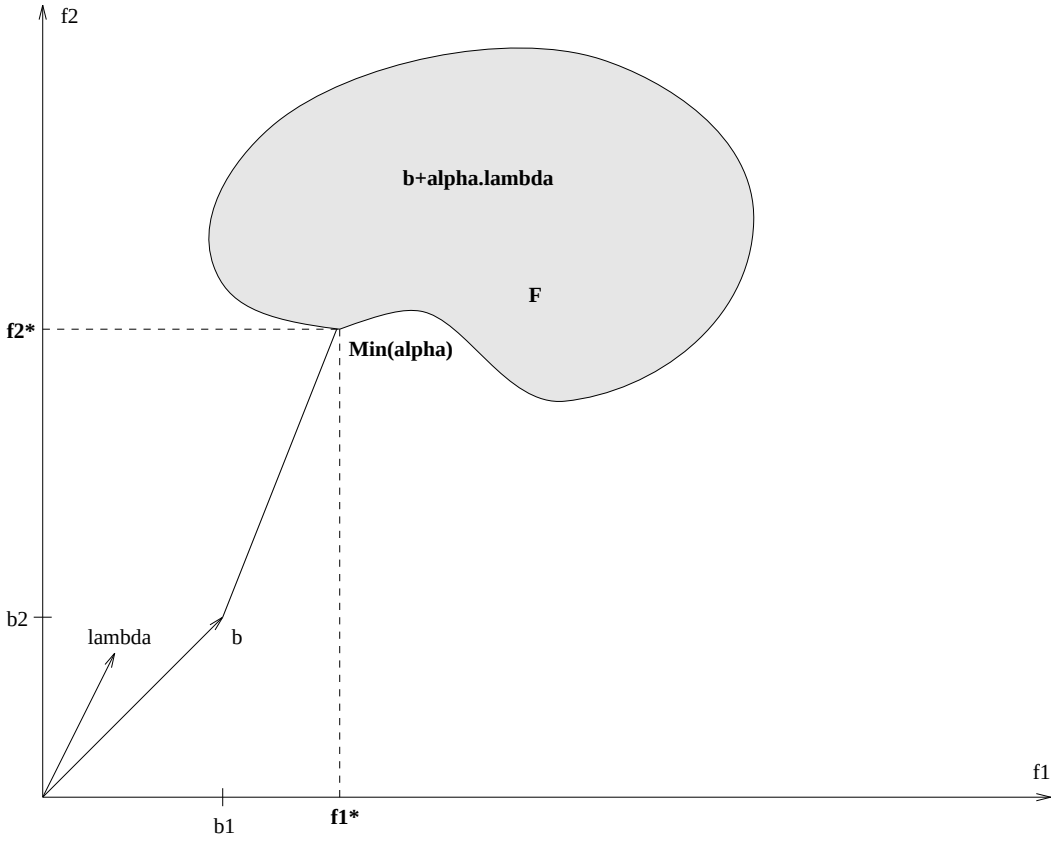


Fig. 10. La méthode de "goal attainment".

une solution réalisable dans l'espace objectif la plus proche de l'origine et dans la direction du vecteur $b + \alpha\lambda$. Si cette solution existe, elle est nécessairement Pareto optimale.

La valeur optimale de α indique si les buts sont atteignables ou non. Une valeur négative de α implique que les buts sont atteignables. Sinon, si $\alpha > 0$, le but n'est pas atteignable.

Le problème formulé ainsi a été résolu en utilisant les algorithmes génétiques [99].

L'inconvénient majeur de cette méthode est l'absence éventuelle de la pression de sélection des solutions générées. Par exemple, si on a deux solutions qui ont la même valeur pour un objectif et des valeurs différentes pour l'autre objectif, elles ont le même coût. L'algorithme de recherche utilisé ne peut donc les différencier dans la résolution du problème.

Analyse de la première classe de méthodes : La transformation du problème multi-objectif en un problème uni-objectif nécessite des connaissances a priori du problème traité. L'optimisation d'un problème uni-objectif peut garantir l'optimalité Pareto de la solution trouvée mais trouve une seule solution. Dans les situations réelles, les décideurs ont généralement besoin de plusieurs alternatives. En effet, si certains objectifs sont bruités ou si des données sont incertaines, ces méthodes ne sont pas efficaces. Ils sont aussi sensibles au paysage de la frontière Pareto (convexité, discontinuité, etc.). L'autre inconvénient de ces méthodes est leur sensibilité par rapport aux poids, aux contraintes ou aux vecteurs de référence. Les solutions obtenues dépendent fortement de ces paramètres. Pour différentes situations, différents paramètres sont utilisés, et le problème doit être résolu plusieurs fois, d'où le coût associé à cette classe d'algorithmes, qui nécessitent parfois l'optimisation indépendante de chacun des objectifs. Dans l'exécution multiple des algorithmes, on espère trouver différentes solutions Pareto optimales.

Dans les deux classes suivantes de méthodes, ces difficultés peuvent être éliminées. Une seule résolution du problème permet de trouver un ensemble de solutions Pareto optimales. Le décideur peut ainsi choisir une solution suivant la situation courante. La connaissance de plusieurs solutions Pareto optimales est utile aussi pour une utilisation future, particulièrement quand la situation change et une nouvelle solution est requise. Nous présentons dans les sections suivantes comment ces méthodes surmontent les difficultés présentées précédemment.

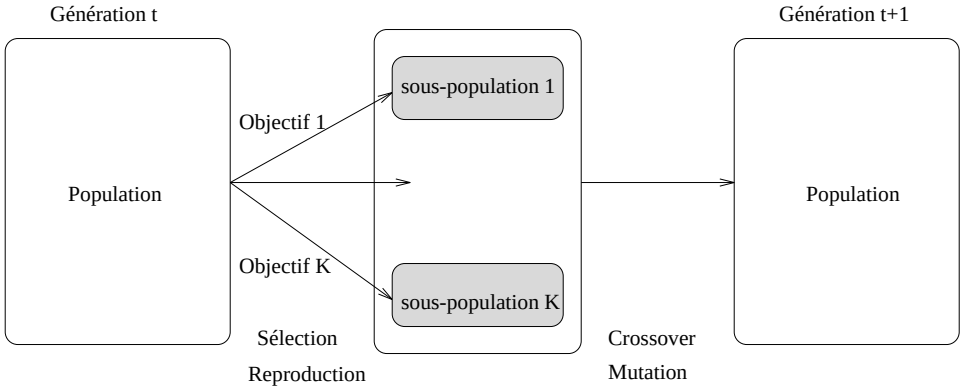


Fig. 11. Sélection parallèle dans l'algorithme VEGA.

VI. Approches non-Pareto

Dans les approches non-Pareto basées sur les populations de solutions, la recherche est réalisée en traitant séparément les différents objectifs non commensurables.

A. Sélection parallèle dans les algorithmes évolutionnaires

Le premier travail consistant à utiliser les AGs pour résoudre des PMO est celui de Schaffer [74]. L'algorithme développé VEGA (Vector Evaluated Genetic Algorithm) sélectionne les individus de la population courante suivant chaque objectif, indépendamment des autres (sélection parallèle). A chaque génération, la population est donc divisée en un nombre de sous-populations qui est égal au nombre d'objectifs de la fonction coût. Chaque sous-population i est sélectionnée suivant l'objectif f_i. L'algorithme VEGA compose la population complète, et applique les opérateurs génétiques (mutation, crossover) (fig.11).

Comme l'algorithme affecte aléatoirement les individus aux sous-populations à chaque génération, un même individu peut être évalué différemment suivant l'objectif qui lui est affecté, d'une génération à l'autre. L'analyse de l'algorithme VEGA a montré que son comportement est le même qu'un algorithme réalisant une agrégation linéaire [69]. Malgré cet effet, Schaffer a reporté de bons résultats dans la recherche des solutions Pareto optimales dans l'optimisation de fonctions réelles. Cependant, il y a une certaine tendance à ignorer les solutions "compromis", et les solutions non-supportées ne peuvent être générées.

L'algorithme VEGA a été utilisé et amélioré par plusieurs auteurs. Surry et Radcliffe l'ont utilisé dans la modélisation des contraintes d'un problème uni-objectif pour éviter l'utilisation des pénalités dans la fonction coût [82]. Leur algorithme traite les contraintes comme des objectifs. La procédure standard de VEGA a été modifiée pour introduire une forme de "ranking" basée sur le nombre de contraintes violées par un individu. D'autres opérateurs de sélection parallèle ont aussi été proposés comme le "n-branch tournament" [45].

B. Sélection lexicographique

Dans cette approche classique, la sélection est réalisée suivant un ordre définie par le décideur. Cet ordre définit l'ordre d'importance des objectifs.

Plusieurs métaheuristiques à base de populations ont été utilisées pour résoudre des PMO en se basant sur la sélection lexicographique [9] :

- **Algorithmes génétiques** : Dans [26], la sélection est basée sur un ordre lexicographique. Elle est réalisée en comparant des paires d'individus ; chaque paire d'individus est comparée suivant l'objectif avec la plus grande priorité. Si le résultat est le même pour cet objectif, alors l'objectif avec la deuxième priorité est utilisé, etc. Comme dans l'algorithme VEGA, ceci correspond à faire une agrégation implicite avec des poids correspondant aux probabilités de choix de chaque objectif.
- **Stratégies évolutionistes** : Une autre approche basée sur une sélection lexicographique a été proposée en utilisant les stratégies évolutionistes [47]. Les individus sont comparés suivant un objectif, qui est choisi de façon aléatoire suivant une probabilité pré-déterminée.

Les méthodes non-Pareto de sélection parallèle et lexicographique reviennent à considérer à chaque génération la moyenne des valeurs associées aux différents objectifs. Dans la stratégie de Schaffer, le résultat attendu correspond, en effet, à une combinaison linéaire des objectifs avec des poids variables suivant la distribution des individus de la population courante [69]. La stratégie de Fourman, quant à elle, correspond à faire une moyenne des rangs et non

pas des valeurs des fonctions coût. Dans les deux cas, des valeurs différentes peuvent être affectées aux individus non-dominés.

L'inconvénient majeur de ces approches est qu'elles tendent à générer des solutions qui sont largement optimisées pour certains objectifs au détriment des autres objectifs. Les solutions "compromis" sont négligées.

VII. Approches Pareto

Les approches Pareto utilisent directement la notion de dominance dans la sélection des solutions générées, contrairement aux autres approches qui utilisent une fonction d'utilité ou traitent séparément les différents objectifs. Cette idée a été introduite initialement dans les AGs par Goldberg [32]. Le principal avantage de ces approches est qu'elles sont capables de générer des solutions Pareto optimales dans les portions concaves de la frontière Pareto.

Les AGs ont été largement utilisés pour la résolution de PMO, étant donné qu'ils travaillent sur une population de solutions. Deux objectifs doivent être pris en compte dans la résolution d'un PMO :

- Converger vers la frontière Pareto : la plupart des travaux de recherche sur l'application des AGs aux PMO se sont concentrés sur l'étape de sélection. Dans cette étape, des *méthodes de ranking* sont appliquées dont le rôle est d'établir un ordre (rank) entre les individus. Cet ordre dépend de la notion de dominance et donc directement de l'optimalité Pareto. Les méthodes de ranking permettent de converger vers les solutions Pareto optimales.
- Trouver des solutions diversifiées dans la frontière Pareto : les *méthodes de maintenance de la diversité*, par la formation de niches écologiques et d'espèces, peuvent être particulièrement utiles pour stabiliser des sous-populations multiples le long de la frontière Pareto.

A. Méthodes de ranking

Plusieurs méthodes de ranking ont été utilisées :

- NSGA (Non-dominated Sorting Genetic Algorithm) : cette première procédure de ranking a été initialement proposée par Goldberg [32] et implémentée par Srinivas et Deb [79]. Tous les individus non dominés de la population possèdent le rang 1. Ces individus sont ensuite enlevés de la population, et l'ensemble suivant d'individus non dominés est identifié et on leur attribue le rang 2 (fig.12). Ce processus est réitéré jusqu'à ce que tous les individus de la population aient un rang. Cette méthode de ranking a été utilisée dans les AGs pour la résolution de plusieurs PMO : arbre recouvrant minimum [102], sac à dos [1], conception de réacteurs [64], distribution de l'eau [36], etc.
  La probabilité de sélection est ensuite affectée à chaque individu en se basant sur le rang, en appliquant par exemple une méthode similaire à celle de Baker proposée pour les problèmes uni-objectifs [3]. La probabilité qu'un individu de rang n de la population courante soit sélectionné est donnée par l'équation suivante :

$$p_n = \frac{S(N+1-R_n) + R_n - 2}{N(N-1)}$$

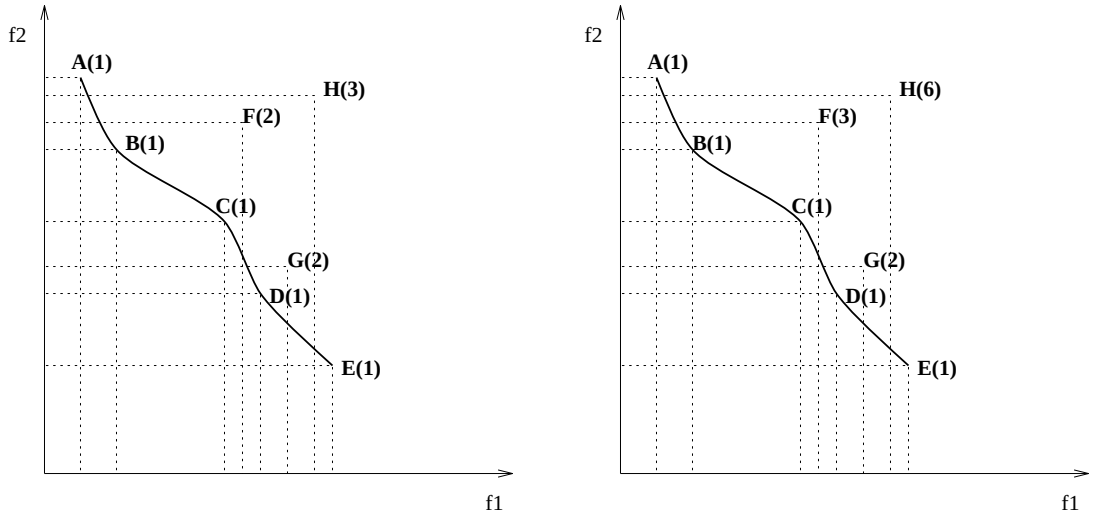
  où N est la taille de la population, S est la pression de sélection, et

$$R_n = 1 + r_n + 2\sum_{i=1}^{n-1} r_i$$

  où r_i est le nombre d'individus de rang i. Une pression de sélection S égale à 1 donne la même probabilité de sélection pour tous les individus ($p_n = \frac{1}{N}$). Pour des valeurs de S supérieures à 2, les individus de dernier rang peuvent avoir des probabilités de sélection négatives. Dans ce cas, ils ne sont jamais sélectionnés.
- NDS (Non Dominated Sorting) : dans cette méthode, le rang d'un individu est le nombre de solutions dominant l'individu plus un [24]. Considérons par exemple un individu c_i à la génération t, qui est dominé par p_i^t individus dans la population courante. Son rang dans la population est donné par :

$$rang(c_i, t) = 1 + p_i^t$$

  Un individu non dominé de la population possède donc le rang 1 (fig.12) [25]. Les rangs associés à la méthode NDS sont toujours supérieurs à ceux de la méthode NSGA. Ce type de ranking induit donc une plus forte pression de sélection, et peut causer une convergence prématurée.
  Cette méthode de ranking a été utilisée dans les AGs pour plusieurs PMO : gestion de containers [87], design d'ailes d'avion [60], et la synthèse de systèmes distribués embarqués [19].
- WAR (Weighted Average Ranking) : les différents coûts de chaque individu sont évalués pour chaque objectif. Une liste de valeurs est établie pour chaque objectif. Ces listes sont alors triées suivant l'ordre décroissant des valeurs, en associant un ordre pour chaque solution et chaque objectif. La moyenne des rangs est finalement calculée pour chaque individu [5].
  Cette méthode de ranking a été utilisée dans les AGs pour l'optimisation de fonctions continues [5].



(a) Ranking de Goldberg (NSGA) (b) Ranking de Fonseca et Fleming (NDS)

Fig. 12. Ordre basé sur l'optimalité Pareto.

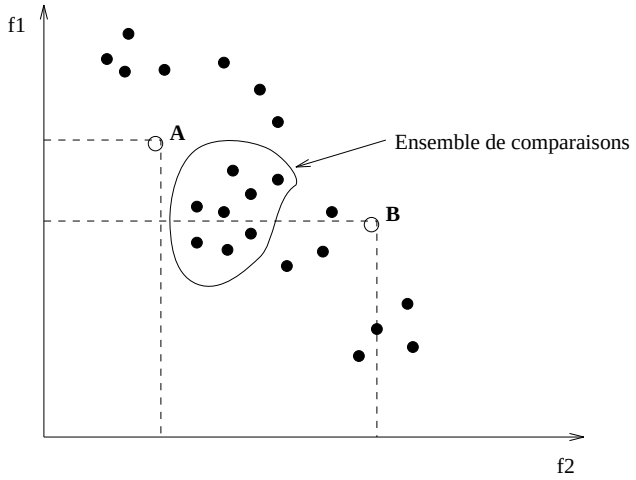


Fig. 13. Sélection basée sur un tournoi. Dans ce cas, l'individu A est non-dominé par l'ensemble de comparaisons, tandis que B est dominé par certains individus de l'ensemble. L'individu A est alors sélectionné pour la reproduction.

A.1 Sélection

Plusieurs méthodes de sélection basées sur la dominance ont été proposées :

- Horn et al. [41] ont proposé une sélection à base de tournoi. A chaque tournoi, deux individus A et B rentrent en compétition par rapport à un ensemble de t_{dom} individus de la population appelé ensemble de comparaison (comparison set). Si le compétiteur A domine tous les individus de l'ensemble et l'autre compétiteur B est dominé par au moins un individu de l'ensemble, alors l'individu A est sélectionné (fig.13). Si les deux individus sont dominés ou non-dominés, la technique basée sur la fonction de partage est adoptée (voir section VII-B.3). Le paramètre t_{dom} contrôle la pression de sélection : si t_{dom} est petit, la pression est faible et le nombre de solutions Pareto dans la population sera réduit. Si t_{dom} est grand, la pression sera forte et une convergence prématurée peut se produire. Cette méthode a été utilisée dans les AGs [12] pour un problème de design de polymère dans l'industrie plastique.
- Tamaki et al. [85] ont proposé une stratégie de réservation Pareto qui correspond à l'élitisme dans le cas uni-objectif. Dans leur méthode, les individus non-dominés sont toujours sauvegardés à la génération suivante. Si le nombre de solutions non-dominées est inférieur à la taille de la population, le reste de la population est produit en appliquant la sélection parallèle de la méthode VEGA. Dans l'autre cas, c-à-d si le nombre d'individus non dominés est supérieur à la taille de la population, les individus de la génération suivante sont sélectionnés parmi les individus non dominés en appliquant la sélection parallèle. Dans une version ultérieure de leur AG, ils ont adopté la technique de partage de fonction dans la sélection des individus non dominés (fig.14).
- Osyczka et Kundu [61] ont proposé une autre méthode de ranking. Le coût associé à un nouvel individu est calculé suivant la distance relative dans l'espace objectif par rapport aux individus non dominés de la population courante.

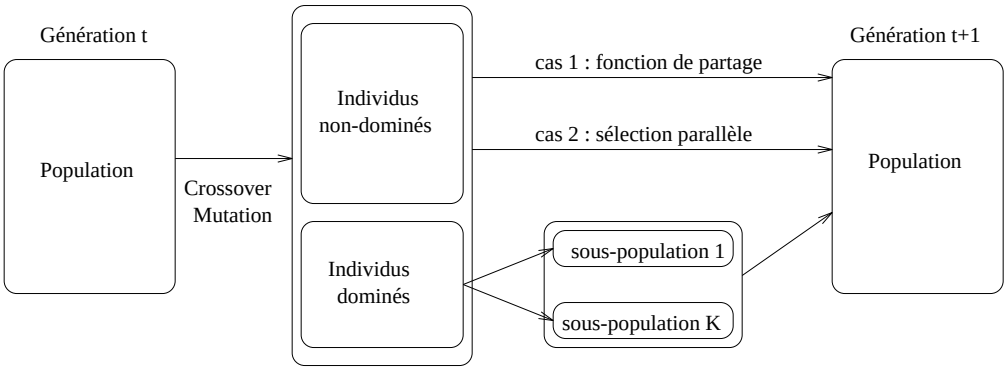


Fig. 14. Stratégie de réservation Pareto. Cas 1 : le nombre d'individus non dominés est supérieur à la taille de la population. Cas 2 : Sinon.

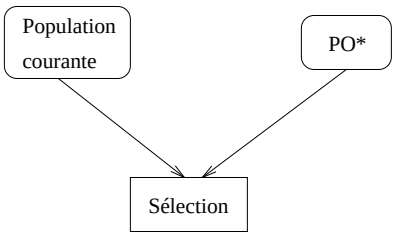


Fig. 15. Elitisme dans la phase de sélection.

A.2 Elitisme

Les travaux récents autour des méthodes Pareto ont montré l'intérêt de l'élitisme pour une meilleure approximation de la frontière Pareto [103][60][58]. L'élitisme consiste à maintenir une population autre que la population courante, qui permet d'archiver toutes les solutions Pareto optimales trouvées durant la recherche. Cette population externe participe dans la définition des opérateurs génétiques (sélection, reproduction, etc.). Dans [103], le rang associé à un individu des deux populations est calculé en se basant sur le nombre de solutions dominées : toutes les solutions non dominées des deux populations ont un rang qui est égal aux nombre de solutions qu'elles dominent. Plus un individu domine des solutions, meilleur est son rang. Ceci permet d'encourager implicitement la diversité.

L'élitisme est très utilisé dans le processus de sélection. Il consiste par exemple à réaliser la sélection des individus aussi bien de la population courante que des solutions non dominées trouvées pendant la recherche [64][58] (fig.15). L'équation VII-A est donc modifiée de la manière suivante :

$$p_n = \frac{N-A}{N}\frac{S(N+1-R_n)+R_n-2}{N(N-1)}$$

où A est le nombre d'individus sélectionnés à partir de l'ensemble PO^* courant.

B. Méthodes de maintenance de la diversité

Dans la résolution de PMO, il est nécessaire que les solutions trouvées soient Pareto optimales, mais aussi qu'elles soient uniformément réparties dans le sous-espace des solutions Pareto optimales. Les méthodes de ranking présentées dans la section précédente permettent d'atteindre le premier objectif. Cependant, le deuxième objectif n'est pas pris en compte. Pour maintenir une diversité dans la population, les méthodes de ranking doivent être utilisées en conjonction avec les techniques de formation de niches écologiques et d'espèces.

Les méthodes de formation de niches ont d'abord été utilisées dans l'optimisation multi-modale. L'optimisation multi-modale a pour but de localiser les différents optima d'un problème uni-objectif donné (fig.16). Dans les AGs, il est connu que la diversité génétique est perdue à cause du processus de sélection stochastique. L'utilisation d'un AG simple converge généralement vers une seule solution ; ce phénomène est connu sous le nom de dérive génétique (genetic drift) [32]. Ainsi, un certain nombre de techniques basées sur les algorithmes évolutionnaires ont été proposées pour la conservation de la diversité.

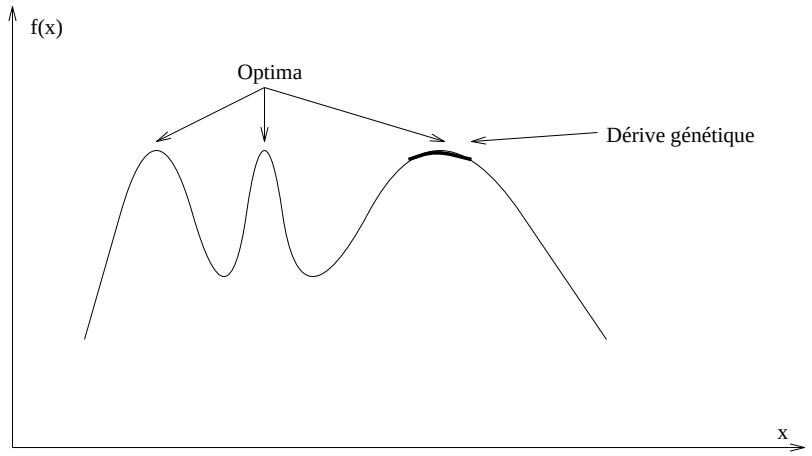


Fig. 16. Optimisation multi-modale.

B.1 Exécutions itératives indépendantes

La première technique est basée sur l'exécution itérative de l'algorithme. Si tous les optima possèdent la même probabilité d'être trouvés, le nombre d'exécutions indépendantes doit être égal à [4] :

$$p \sum_{i=1}^{p} \frac{1}{i} \approx p(\alpha + \log p)$$

où p est le nombre d'optima, et $\alpha \approx 0.577$ est la constante d'Euler.

Cependant, dans la plupart des applications réelles, les optima n'ont pas la même probabilité d'être trouvés. Le nombre d'exécutions indépendantes doit donc être beaucoup plus important. Dans ce cas, l'exécution parallèle d'un AG sur des populations indépendantes peut s'avérer utile. Toujours est-il que cette méthode n'a pas donné de bons résultats pour des problèmes réels.

B.2 Niching séquentiel

Dans le niching séquentiel, la localisation de multiples niches se fait de manière séquentielle, à l'aide d'une exécution itérative de l'algorithme. Dans [4], Beasly, Bull et Martin ont décrit une méthode, basée sur le niching séquentiel, pour l'optimisation de fonctions multi-modales, qui évitent les inconvénients des méthodes d'exécutions itératives indépendantes. Leur stratégie est basée sur l'idée suivante : une fois qu'un optimum est trouvé, la fonction d'évaluation est modifiée dans le but de pénaliser, dans le processus de recherche, l'optimum déjà trouvé. Les étapes principales de l'algorithme sont présentées dans la figure 17.

1. Initialisation : affecter à la fonction coût modifiée la fonction coût originale.
2. Exécuter l'AG en utilisant la fonction coût modifiée,
et en sauvegardant la meilleure solution trouvée durant la recherche.
3. Mettre à jour la fonction coût modifiée pour éviter la recherche
dans les régions de la meilleure solution trouvée précédemment.
4. Si toutes les solutions n'ont pas encore été trouvées, alors retour à l'étape 2.

Fig. 17. Niching séquentiel.

L'inconvénient de cette approche est qu'elle modifie le paysage du problème original.

D'autres méthodes avancées ont donc été proposées pour favoriser la formation de niches dans les algorithmes génétiques. Ces méthodes de diversification sont basées sur le niching parallèle, comme par exemple les fonctions de partage et le crowding. Dans le niching parallèle, la formation et la maintenance des niches se font à l'aide d'une seule exécution de l'algorithme. Ces méthodes posent un certain nombre de questions :

- Dans quel espace la distance est-elle mesurée ? : comme pour les mesures de similarités entre individus, l'espace peut être l'espace du domaine des fonctions objectifs (espace objectif) [25][41], ou bien l'espace des variables de décision (espace de décision) [79].
- Quelle métrique utiliser ?
- Comment dégrader le coût quand il est vectoriel et non pas scalaire ?
- Comment estimer la taille des niches ?

B.3 Fonction de partage

Goldberg et Richardson ont présenté une méthode basée explicitement sur la métaphore de partage pour induire des niches écologiques [33]. Une *fonction de partage* (sharing function) est définie afin de déterminer le voisinage et le degré de partage pour chaque individu de la population. La méthode permet la formation de sous-populations stables (species), dans le sens que l'algorithme permet d'explorer plusieurs optima en parallèle. Le partage dégrade le coût d'un individu par rapport au nombre d'individus similaires dans la population. La désirabilité (fitness) d'un individu x est égale à la fonction coût f divisée par son compteur de niche.

$$f^{'}(x) = \frac{f(x)}{m(x)}$$

où $m(x)$ retourne le compteur de niche pour un individu x. Le compteur de niche est la somme des fonctions de partage (sh) entre l'individu x et tous les autres individus de la population.

$$m(x) = \sum_{y \in pop} sh(dist(x,y))$$

Dans la formule précédente, la somme sur tous les individus y de la population inclue l'individu x. Par conséquent, si l'individu x est dans sa propre niche, son coût ne va pas décroître. Sinon, la fonction est décrémentée proportionnellement au nombre et à la distribution des individus voisins. Quand plusieurs individus sont dans le même voisinage, ils contribuent au compteur de partage d'un individu.

La fonction de partage sh possède les propriétés suivantes :
1. $0 \leq sh(dist(x,y)) \leq 1$
2. $sh(0) = 1$
3. $lim_{dist(x,y) \to \infty} sh(dist(x,y)) = 0$

Il existe plusieurs fonctions de partage qui satisfont les conditions précédentes. La fonction de partage classique suggérée dans [33] est :

$$sh(dist(x,y)) = \begin{cases} 1 - (\frac{dist(x,y)}{\gamma_{sh}})^{\alpha} & \text{Si } dist(x,y) < \gamma_{sh} \\ 0 & \text{Sinon} \end{cases}$$

où γ_{sh} et α sont des constantes. La constante γ_{sh} spécifie le seuil de dissimilarité (taille des niches) ; si la distance entre deux individus est supérieure à γ_{sh}, ils n'affectent pas leur fonction de partage respectivement. La constante α, généralement initialisée à 1, permet de réguler la forme de la fonction de partage.

La fonction de partage sh dépend de la distance $dist$ entre deux individus de la population. Elle retourne 1 si les deux individus sont identiques, 0 s'ils sont différents (à partir d'un seuil donné), et une valeur intermédiaire pour des niveaux de similarité intermédiaires.

La distance $dist$ peut être dans l'espace de décision ou dans l'espace objectif [24], et dépend généralement du problème traité. Le maintien de la diversité dans l'espace objectif peut ne pas générer une diversité dans l'espace de décision, qui peut être importante pour le décideur. L'algorithme peut ne pas trouver de multiples solutions pour des problèmes où différentes solutions Pareto optimales correspondent à la même valeur pour la fonction objectif.

Une méthode combinant une distance génotypique (distance de Hamming) et une distance phénotypique (espace objectif) a été utilisée dans [71][53]. Soient $d_1(x_i, x_j)$ la distance de Hamming entre deux individus x_i et x_j, et $d_2(x_i, x_j)$ la mesure de la distance dans l'espace objectif. La distance d_2 représente la somme des différences de coûts correspondant à chaque dimension :

$$d_2(x_i, x_j) = \sum_{k=1}^{n} |f_k(x_i) - f_k(x_j)|$$

La fonction de partage a été définie de la façon suivante :

$$sh(x_i, x_j) = \begin{cases} 1 - \frac{d_1(x_i,x_j)}{\gamma_1} & \text{Si } d_1(x_i,x_j) < \gamma_1, d_2(x_i,x_j) \geq \gamma_2 \\ 1 - \frac{d_2(x_i,x_j)}{\gamma_1} & \text{Si } d_1(x_i,x_j) \geq \gamma_1, d_2(x_i,x_j) < \gamma_2 \\ 1 - \frac{d_1(x_i,x_j) d_2(x_i,x_j)}{\gamma_1 \gamma_2} & \text{Si } d_1(x_i,x_j) < \gamma_1, d_2(x_i,x_j) < \gamma_2 \\ 0 & \text{Sinon} \end{cases}$$

où γ_1 et γ_2 correspondent à la taille des niches dans les deux espaces (décision et objectifs).

D'autres techniques hybridant le partage dans l'espace de décision et l'espace objectif sont proposées dans [40].

La question qui reste posée concerne la taille des niches, c-à-d l'estimation du paramètre γ_{sh}. Les performances des algorithmes sont très sensibles à ce paramètre.

Dans [18], Deb et Goldberg ont proposé une méthode pour l'estimation du paramètre γ_{sh} dans l'espace phénotypique et génotypique. Dans l'espace phénotypique, la distance entre deux individus est la distance euclidienne dans un espace à n dimensions :

$$d_{ij} = \sqrt{\sum_{k=1}^{n} (f_{ki} - f_{kj})^2}$$

où f_{ki} et f_{kj} représentent les valeurs de l'objectif f_k associées aux individus i et j. Cette distance peut être normalisée comme proposé dans [79] :

$$d_{ij} = \sqrt{\sum_{k=1}^{n} (\frac{f_{ki} - f_{kj}}{max_k - min_k})^2}$$

où max_k et min_k représentent la borne maximale et minimale pour l'objectif f_k.

Pour estimer γ_{sh}, l'expression suivante est utilisée :

$$\gamma_{sh} = \frac{r}{\sqrt[n]{q}} = \frac{\sqrt{\sum_{k=1}^{n} (f_{ki} - f_{kj})^2}}{\sqrt[n]{2q}}$$

où r est le volume d'une hypersphère de rayon γ_{sh} dans un espace à dimension n, et q est le nombre d'optima qu'on désire trouver avec l'algorithme génétique.

Dans l'espace génotypique, d_{ij} est définie comme la distance de Hamming entre les individus et γ_{sh} est le nombre maximum de bits différents permis entre deux individus formant des niches séparées dans la population. Les expériences réalisées montrent que le partage est meilleur que le crowding, et que le partage dans l'espace phénotypique est meilleur que dans l'espace génotypique.

L'inconvénient de cette approche est la nécessité d'avoir une idée sur la séparation minimale des "optima" désirés.

B.4 Crowding

Holland a été le premier à suggérer l'utilisation de l'opérateur de "crowding" dans la phase de remplacement des AGs [39], pour identifier les situations dans lesquelles de plus en plus d'individus dominent les niches écologiques. Une première implémentation de cet opérateur a été réalisée par DeJong. Dans la reproduction d'un nouvel individu, l'opérateur consiste à remplacer l'individu existant le plus semblable à l'individu généré, et non pas les parents comme dans les AGs standards.

Par exemple, le crowding déterministe défini dans la figure 18 réalise un tournoi qui force la compétition entre individus semblables [54].

- Sélectionner aléatoirement 2 parents p_1 et p_2 ;
- Réaliser un crossover produisant c_1 et c_2 ;
- Appliquer la mutation produisant c_1' et c_2' ;
- **Si** $(d(p_1, c_1') + d(p_2, c_2')) \leq (d(p_1, c_2') + d(p_2, c_1'))$ **Alors**
  $f(c_1') > f(p_1)$ Alors remplacer p_1 par c_1' ;
  $f(c_2') > f(p_2)$ Alors remplacer p_2 par c_2' ;
- **Sinon**
  $f(c_2') > f(p_1)$ Alors remplacer p_1 par c_2' ;
  $f(c_1') > f(p_2)$ Alors remplacer p_2 par c_1' ;

Fig. 18. Crowding déterministe.

A notre connaissance, aucun algorithme n'a utilisé un tel opérateur dans la résolution de PMO.

Une comparaison des différentes méthodes de maintien de niches (séquentielle et parallèle) est effectuée dans [55]. Les résultats indiquent que les méthodes parallèles sont meilleures (partage, crowding) que les méthodes séquentielles.

B.5 Modèles parallèles

Peu de travaux ont été réalisés autour des modèles parallèles pour l'optimisation multi-objectif. La plupart des travaux ont porté sur les algorithmes génétiques. Les modèles parallèles d'algorithmes génétiques peuvent être décomposés en deux classes [84] :

- Le modèle insulaire dans lequel plusieurs sous-populations communiquent par migration d'individus (island model).
- Le modèle cellulaire qui utilise une seule population d'individus faiblement connectés, et où la sélection est locale.

Grosso, dans sa thèse, a montré que le modèle insulaire des AGs basé sur la migration entre plusieurs sous-populations est intéressant pour la formation de niches écologiques [34]. Il a pu montrer l'avantage des taux de migration intermédiaires par rapport à des sous-populations isolées (pas de migration) ou complètement connectées (modèle standard).

Dans [71], un modèle cellulaire a été proposé pour résoudre un PMO. La population est projetée sur un tore. Chaque individu sélectionne aléatoirement un individu voisin et se recombine avec lui. L'individu généré remplace l'individu local s'il est meilleur (élitisme). L'algorithme est décrit dans la figure 19.

1. Générer une population initiale de façon aléatoire $x = (x_1, ..., x_M)$ et la placer sur un tore
2. **Pour** chaque individu x_i **Faire**
   Choisir aléatoirement un individu voisin x_j de x_i
   Recombiner x_i et x_j avec un crossover pour générer y_i
3. **Pour** chaque individu x_i **Faire**
   Si y_i est meilleur que x_i **Alors** $x_i := y_i$
   Appliquer la mutation à x_i
4. Aller à 2

Fig. 19. Modèle cellulaire parallèle pour l'optimisation multi-objectif.

Le modèle cellulaire permet une meilleure diversité que le modèle séquentiel, où la pression de sélection est plus forte. Le coût de calcul du rang d'un individu est réduit, étant donné qu'une seule comparaison est suffisante dans la phase de sélection. Un des inconvénients de cette approche est que certaines régions du tore sont inutiles, dans le sens que si deux solutions en compétition dans la même niche sont physiquement séparées dans la population, elles ne seront jamais en compétition, et elles seront donc dans la solution finale.

Le parallélisme a aussi été utilisé comme moyen d'accélérer le processus de recherche. Dans [45], l'évaluation des fonctions objectifs est réalisée en parallèle sur une machine IBM SP2 de 64 processeurs utilisant l'environnement MPI. Etant donné que le coût de l'évaluation d'une solution dépend de sa configuration, des mécanismes de régulation dynamique de charge ont été développés pour équilibrer la charge des processeurs.

B.6 Restriction de voisinage

D'autres travaux proposés pour le maintien de la diversité dans une population sont basés sur la restriction de voisinage. L'idée est de permettre à deux individus de se reproduire s'ils sont similaires. Ceci induit la formation de différentes "espèces" (mating groups) dans la population [29]. Par exemple, dans [51], les individus sont projetés sur les éléments d'une matrice de dimension (n-1), n étant le nombre d'objectifs. Le voisinage est restreint aux individus se trouvant dans un rayon inférieur à r, r étant fixé avant l'exécution de l'algorithme.

D'autres travaux ont proposé le contraire, c-à-d qu'ils ne permettent pas la reproduction entre individus qui sont similaires, pour prévenir l'inceste [22].

D'autres approches proposées par Spears [78] et Hajela [35] utilisent conjointement le principe du partage de fonction et la restriction d'appariement.

VIII. Evaluation de performances et structure de la frontière Pareto

Dans cette section, l'évaluation de performances des métaheuristiques et la caractérisation de la structure de la frontière Pareto sont abordées.

A. Evaluation des performances

Peu de travaux présentés dans la littérature effectuent une évaluation quantitative des méthodes proposées. Afin d'évaluer les performances des métaheuristiques, on peut se référer à une représentation exacte de l'ensemble des solutions PO. Pour certains problèmes (cheminement, flots, etc.) ou pour des problèmes de petite taille (sac-à-dos bi-critère, ordonnancement, etc.), celle-ci peut être obtenue.

A.1 Ensemble PO connu

La première mesure AE (Absolute Efficiency) permet de calculer la proportion des solutions Pareto de l'ensemble approximé PO^* [88] :

$$AE = \frac{|PO^* \bigcap PO|}{|PO|}$$

Cette mesure a été utilisée dans plusieurs travaux, où l'ensemble PO est trouvé par une méthode énumérative [102][43][31], Branch and Bound [88], etc.

Une solution appartenant à PO^* sans être optimale Pareto n'est pas nécessairement une mauvaise solution. Il est donc intéressant de calculer la distance entre l'ensemble PO et PO^* [14]. Plus petite est la distance, meilleure est la

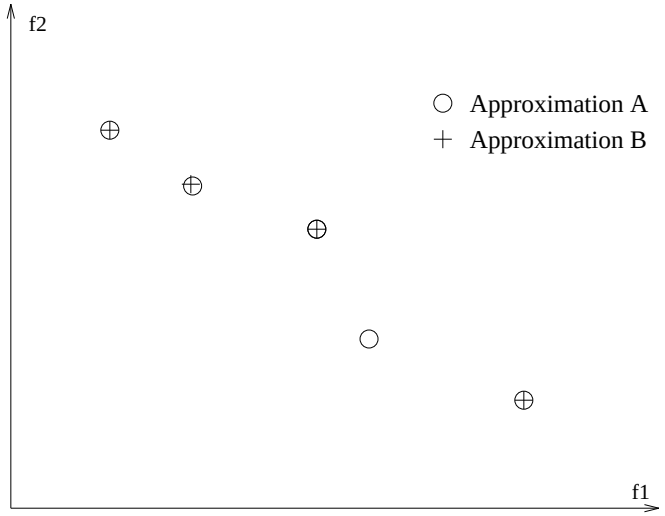


Fig. 20. Approximation faiblement dominée.

qualité de l'algorithme. Soit $d(x, y)$ une distance entre deux solutions dans l'espace objectif (norme L_1 de Tchebycheff) :

$$d(x, y) = \sum_{i=1}^{n} \lambda_i |f_i(x) - f_j(y)|$$

où λ_i est un poids qui permet de normaliser les différents critères. D'autres types de distances peuvent être choisis. La distance entre une solution de PO^* et une solution quelconque y peut donc être définie de la manière suivante :

$$d(PO^*, y) = min\ d(x, y),\ x \in PO^*$$

Etant données ces définitions, nous pouvons considérer les différentes mesures quantitatives suivantes [88] :

- la plus mauvaise distance entre les ensembles PO^* et PO :

$$WD = max\ d(PO^*, y),\ y \in PO$$

- la distance moyenne entre les ensembles PO^* et PO :

$$MD = \frac{\sum_{y \in PO} d(PO^*, y)}{|PO|}$$

- une mesure d'uniformité de l'ensemble PO^* :

$$DIV = \frac{WD}{MD}$$

Ces mesures ont été utilisées dans [88] pour évaluer les performances d'un recuit simulé dans la résolution du problème du sac à dos multi-objectif.

A.2 Ensemble PO inconnu

Pour la plupart des problèmes, il est insoluble d'avoir une représentation analytique ou une énumération exacte de l'espace Pareto. On doit se contenter d'une approximation de l'ensemble des solutions PO. Il est donc difficile de mesurer les performances d'un algorithme d'optimisation multi-objectif.

Une famille de relations de comparaison ont été introduites dans [37] :

- Faible : une approximation A domine faiblement une approximation B, si

$$A \neq B, \text{ et } ND(A \bigcup B) = A$$

où $ND(S)$ dénote l'ensemble des solutions non-dominées de l'ensemble S.
i.e. si pour chaque solution $x_B \in B$ il existe une solution $x_A \in A$ qui est égale ou domine x_B, et au moins une solution de A n'est pas contenu dans B (fig.20).

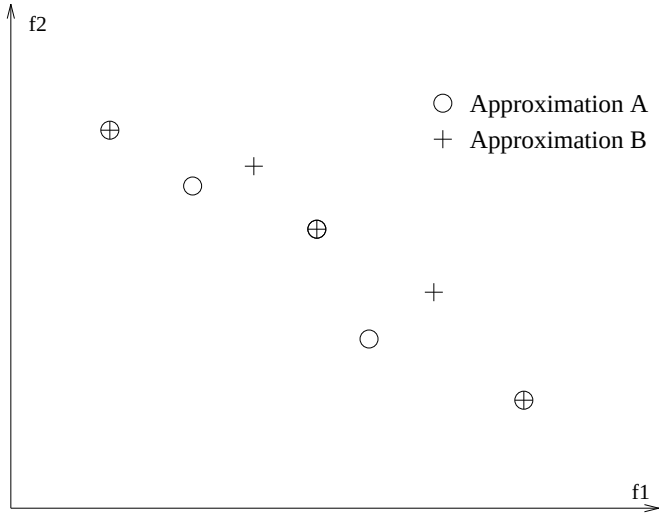


Fig. 21. Approximation fortement dominée.

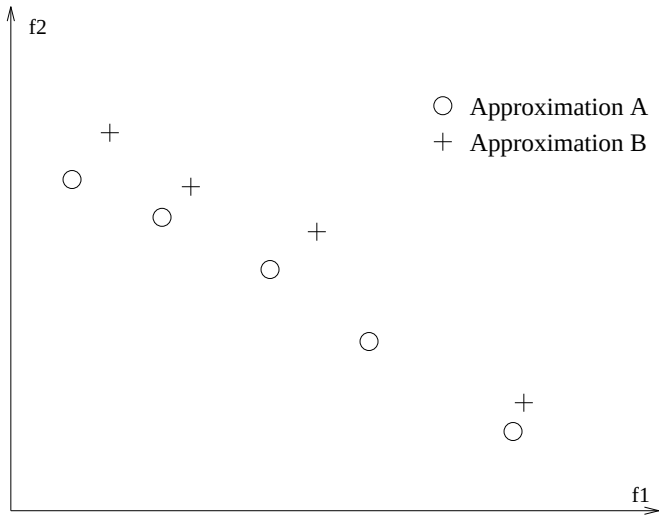


Fig. 22. Approximation totalement dominée.

- Forte : une approximation A domine fortement une approximation B, si

$$ND(A \bigcup B) = A, \text{ et } B - ND(A \bigcup B) \neq \emptyset$$

  i.e. pour chaque solution $x_B \in B$, il existe une solution $x_A \in A$ qui est égale ou domine x_B, et au moins une solution $x_B \in B$ est dominée par une solution $x_A \in A$ (fig.21).
- Totale : une approximation A domine totalement une approximation B, si

$$ND(A \bigcup B) = A, \text{ et } B \bigcap ND(A \bigcup B) = \emptyset$$

  i.e. si chaque solution $x_B \in B$ est dominée par une solution $x_A \in A$.

Chacune des relations présentées définit un ordre partiel dans l'ensemble des approximations. De plus, elles permettent seulement une comparaison qualitative des approximations. La figure 23 montre un exemple d'approximations qui sont incomparables.

La mesure RE (Relative Efficiency) permet de comparer deux méthodes approximatives M_1 et M_2, en mesurant le nombre de solutions Pareto trouvées par M_1 dominées par celles trouvées par M_2.

Cette mesure a été utilisée dans [1] pour comparer les performances d'un algorithme génétique, de la recherche tabou et d'un algorithme hybridant les deux méthodes. Elle a aussi été utilisée pour comparer différents algorithmes génétiques [43].

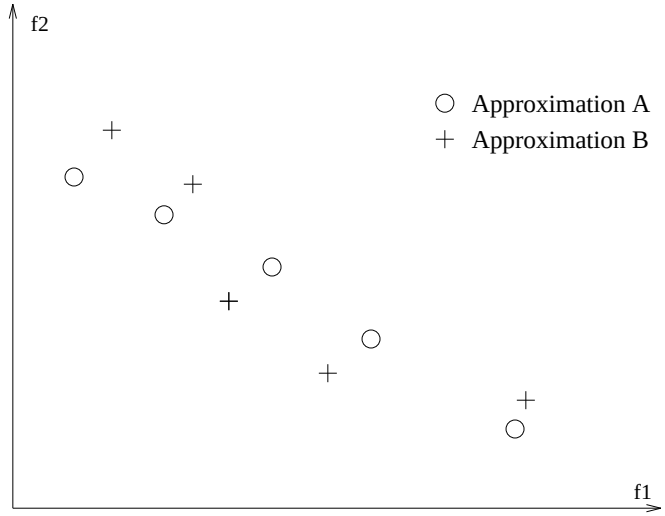


Fig. 23. Approximations incomparables par rapport aux relations faible, complète et totale.

Nous introduisons deux nouveaux indicateurs de performances permettant de comparer deux métaheuristiques notées M_1 et M_2 : la *contribution* et *l'entropie*. Soit PO_i^* l'ensemble Pareto trouvé par la métaheuristique M_i, et C l'ensemble des solutions Pareto communes à M_1 et M_2 (Figure ??).

$$C = PO_1^* \bigcap PO_2^*$$

Soit W_1 (resp. W_2) l'ensemble des solutions de PO_1^* (resp. PO_2^*) qui dominent des solutions dans PO_2^* (resp. PO_1^*).

Soit L_1 (resp. L_2) l'ensemble des solutions dans PO_1^* (resp. PO_2^*) dominées par des solutions dans PO_2^* (resp. PO_1^*).

L'ensemble des solutions dans PO_1^* (resp. PO_2^*) n'ayant pas de relation de dominance avec toutes les solutions de PO_2^* (resp. PO_1^*) est alors :

$$N_1 = PO_1^* - (C \bigcup W_1 \bigcup L_1)$$

Soit PO^* l'ensemble des solutions Pareto optimales trouvées par les deux métaheuristiques M_1 et M_2 :

$$PO^* = C \bigcup W_1 \bigcup N_1 \bigcup W_2 \bigcup N_2$$

La contribution de la métaheuristique M_1 relativement à M_2, dénotée par $Cont(M_1/M_2)$, est le ratio des solutions dans PO^* produites par M_1 :

$$Cont(M_1/M_2) = \frac{\frac{|C|}{2} + |W_1| + |N_1|}{|C| + |W_1| + |N_1| + |W_2| + |N_2|} = \frac{\frac{|C|}{2} + |W_1| + |N_1|}{|PO^*|}$$

Notons que si les deux métaheuristiques produisent les mêmes solutions alors $Cont(M_1/M_2) = Cont(M_2/M_1) = \frac{1}{2}$. Si toutes les solutions produites par M_2 sont dominées par les solutions produites par M_1 alors $Cont(M_2/M_1) = 0$.

L'utilisation de la contribution pour évaluer les performances d'une métaheuristique n'est pas suffisante. Nous avons introduit un autre indicateur, complémentaire à la contribution, permettant d'évaluer la diversité d'un front Pareto : *l'entropie*.

L'entropie $E(PO_1^*/PO_2^*)$ mesure la diversité d'un front Pareto PO_1^* par rapport à un front PO_2^*. L'espace objectif à n dimensions, où n représente le nombre d'objectifs, est partitionné. Pour chaque partition i qui contient au moins un point non dominé de $PO^* = PO_1^* \bigcup PO_2^*$, on calcule le nombre de points n_i appartenant à PO_1^*. L'entropie $E(PO_1^*/PO_2^*)$ est alors :

$$E(PO_1^*/PO_2^*) = \frac{-1}{log(c)} \sum_{i=1}^{c} (\frac{n_i}{c}.log\frac{n_i}{c})$$

où c représente le nombre de partitions de PO^*. Plus l'entropie est proche de 1, meilleure est la diversité de PO_1^* par rapport à PO_2^*.

L'utilisation des deux métriques présentées permet ainsi la comparaison de différentes métaheuristiques et l'ajustement des paramètres d'une métaheuristique donnée. Les mesures présentées ne tiennent pas compte des préférences du décideur.

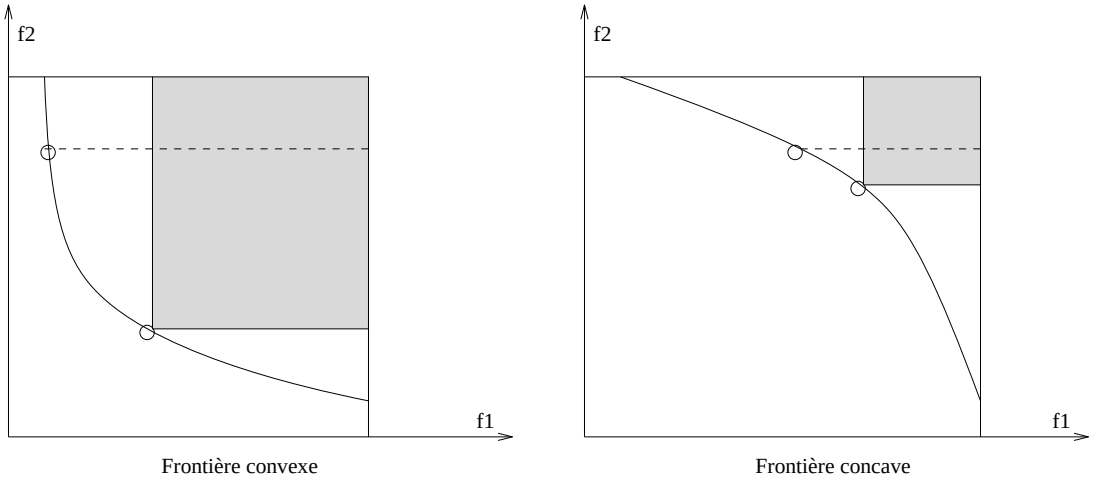


Fig. 24. Méthodes de ranking et convexité de la frontière Pareto.

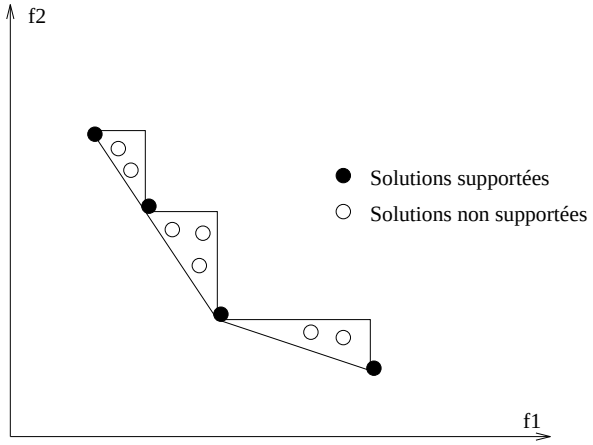


Fig. 25. Solutions supportées et non-supportées.

L'utilisation des métriques présentées permet la comparaison de différentes métaheuristiques et l'ajustement des paramètres d'une métaheuristique donnée. Cependant, les mesures présentées ne tiennent pas compte des préférences du décideur.

B. Caractérisation du paysage de la frontière Pareto

La structure du paysage de la frontière Pareto peut fournir des informations utiles au décideur. Elle permet aussi de dériver des métaheuristiques efficaces pour la résolution du problème.

La structure de la frontière Pareto peut être caractérisée de différentes manières :

- **Structure convexe / concave :** les performances des métaheuristiques diffèrent suivant que la structure de la frontière Pareto est convexe ou concave. Par exemple, l'utilisation des méthodes d'agrégation est plus efficace lorsque la structure est convexe que dans le cas où la frontière est concave.
  Les procédures de ranking dans les algorithmes génétiques peuvent avoir un effet considérable sur les résultats suivant que la structure de la frontière Pareto est convexe ou concave [17]. Dans certaines implémentations d'AGs, le rang d'un individu est proportionnel au nombre de solutions qu'il domine [103]. La figure 24 montre comment cette procédure de ranking favorise les solutions Pareto intermédiaires, dans le cas où la frontière Pareto est convexe. Ceci fait que l'algorithme a tendance à trouver plus de solutions intermédiaires que les solutions à l'extrémité de la frontière. Cette tendance ne se présente pas dans le cas où la frontière Pareto est concave.
  Pour mesurer la convexité de la frontière Pareto, il suffit de faire une distinction entre les solutions supportées et les solutions non supportées. Soit SPO l'ensemble des solutions supportées. La figure 25 représente un exemple dans un espace à deux dimensions. Les solutions non supportées appartiennent aux triangles construits par deux solutions supportées voisines. Ces triangles sont donc des zones d'existence de solutions non supportées.
  Une mesure $CONV$ pour caractériser la convexité de l'ensemble, est la proportion de solutions de PO qui sont

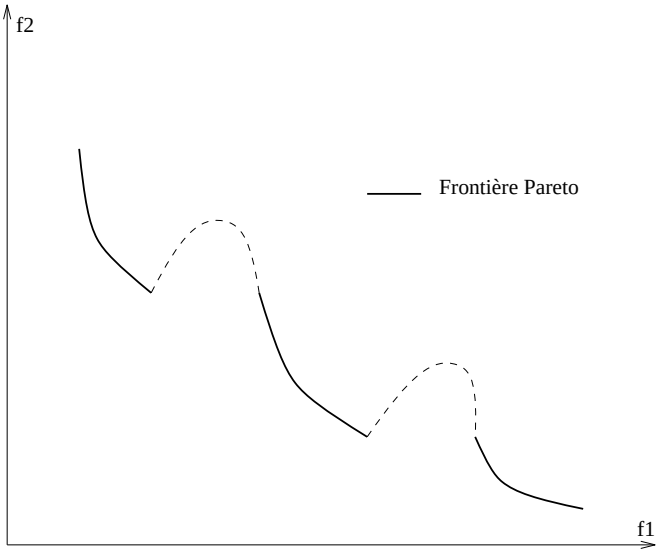


Fig. 26. Frontière Pareto discontinue.

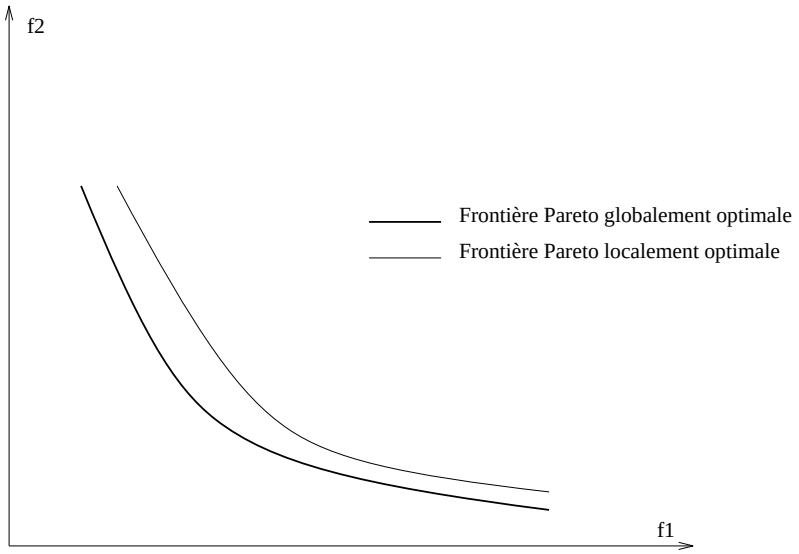


Fig. 27. Frontière Pareto local et global.

supportées :

$$CONV = \frac{|SPO|}{|PO|}$$

- **Structure continue / discontinue :** la discontinuité de la frontière Pareto pose un problème au niveau de la diversité des solutions trouvées par un algorithme génétique (fig.26).
  Nous introduisons la notion de *graphe efficace*, dont l'ensemble des sommets représente les solutions Pareto, et l'ensemble des arêtes représente le voisinage entre les solutions. L'efficacité des métaheuristiques basées sur la recherche locale dépendent de la connexité du graphe efficace [20].
  La distribution non uniforme des solution Pareto optimales peut aussi poser des problèmes au niveau de la maintenance de la diversité au cours de la recherche. Dans ce cas, même si des solutions Pareto peuvent être trouvées dans chacune des régions Pareto, la compétition entre les solutions n'encourage pas l'exploration de l'ensemble des solutions. Une taille adaptative des niches peut être envisagée.
- **Structure multi-modale :** la multi-modalité de la frontière Pareto découle de l'existence de solutions localement Pareto optimales.
  La multi-modalité pose des difficultés au niveau de la convergence vers l'ensemble Pareto optimal exact PO (fig.27). Des techniques permettant de sortir des optima locaux doivent être utilisées, comme la recherche tabou, le recuit simulé, etc.

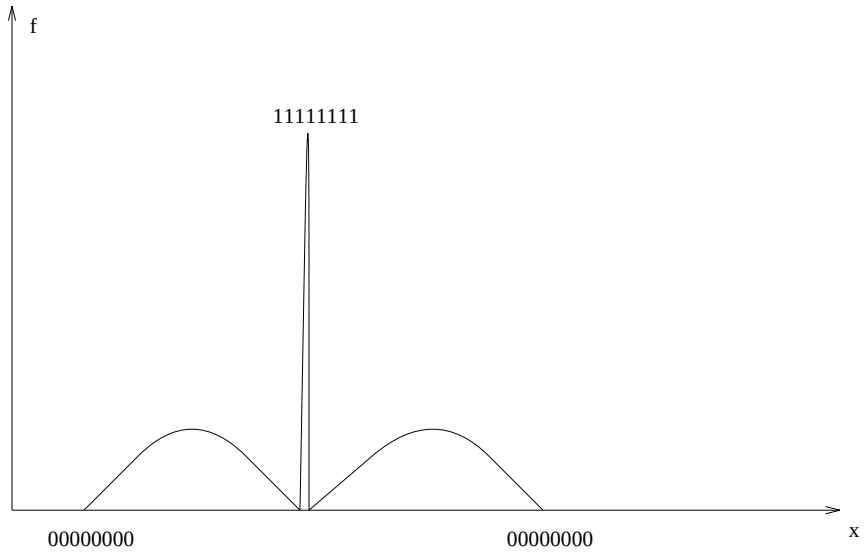


Fig. 28. Problème déceptif pour la recherche locale et facile pour les algorithmes génétiques.

Il peut exister des problèmes où la structure de l'espace de recherche est *"plate"*. La difficulté de ces problèmes vient du fait qu'il n'y a pas d'informations acquises pendant la recherche sur la localisation des solutions Pareto optimales. Les solutions optimales sont isolées dans l'espace de recherche.

- **Structure déceptive :** il existe des problèmes qui sont faciles pour un algorithme de recherche donné et difficiles pour d'autres algorithmes. Ci-dessous un problème bi-critère déceptif pour la recherche locale et facile pour les algorithmes génétiques [52]. Le premier critère est :

$$f_1(x \neq 1111...) = |2.x_1 - x_2|$$

$$f_1(1111....) = x_1.x_2$$

où x est une chaine binaire, x_1 représente le nombre de bits égales à 1 dans la première moitié de la chaine et x_2 le nombre de bits à 1 dans l'autre moitié de la chaine. Le coût croit quand le nombre de 1 augmente jusqu'à la moitié de la chaine, et décroit quand le nombre de 1 augmente dans la deuxième moitié de la chaine.
Le second critère est le suivant :

$$f_2(x \neq 1111...) = |2.x_2 - x_1|$$

$$f_2(1111....) = x_1.x_2$$

Ce problème est représenté par la figure 28. Une fois que l'algorithme génétique a trouvé les optima locaux, le crossover peut produire en une seule opération l'optimum global, en supposant que la population contient des représentants des deux optima locaux. Ceci montre l'intérêt du crossover pour des problèmes multi-critères où il s'agit de trouver un compromis entre plusieurs critères. Des opérateurs de niching peuvent être utilisés pour encourager la diversité des optima locaux. Pour ce problème, on peut remarquer que la restriction de voisinage n'est pas satisfaisante.

IX. Conclusion et perspectives

L'optimisation multi-objectif est sans doute un axe de recherche primordial pour les scientifiques et les ingénieurs, non seulement à cause de la nature multi-critère des problèmes réels, mais aussi parce qu'il reste plusieurs questions ouvertes dans ce domaine.

Cet article présente un état de l'art et une classification des métaheuristiques pour la résolution de problèmes d'optimisation combinatoire multi-critères. Une analyse critique de chaque classe de méthodes est aussi réalisée. Dans le passé, la plupart des articles de synthèse sur les méthodes d'optimisation multi-objectif concernaient la programmation mathématique [80][7]. Des articles de synthèse sur l'application de métaheuristiques ont vu récemment le jour, mais portent sur des méthodes spécifiques comme les algorithmes évolutionnaires [9][25][16] qui ont reçu un intérêt croissant ces dernières années. D'autres métaheuristiques ont été utilisées pour résoudre des PMO : recuit simulé, recherche tabou, colonies de fourmis. A notre connaissance, les métaheuristiques suivantes n'ont pas encore été explorées : GRASP, guided local search, scatter search, acceptation à seuil, méthodes de bruitage.

Dans cet article, nous avons mis l'accent sur quelques axes de recherche non encore explorés, et qui, à notre avis, possèdent un intérêt primordial dans la résolution de problèmes multi-critères :

- Hybridation de méthodes : Une comparaison des ensembles des solutions Pareto trouvés par différentes méthodes d'optimisation montre que ces méthodes ne fournissent pas le même ensemble de solutions efficaces. Les différentes métaheuristiques sont donc complémentaires et non pas compétitives. Ceci montre l'intérêt de l'hybridation de métaheuristiques pour le résolution de PMO.
- Conception et réalisation parallèle d'algorithmes : Le parallélisme peut être utilisé comme moyen d'accélérer le processus de recherche. En effet, on peut utiliser les modèles parallèles des métaheuristiques appliquées à la résolution de problèmes d'optimisation mono-objectif. De plus, la phase d'évaluation des différents objectifs peut facilement être parallélisé.

  Le parallélisme est aussi un moyen pour améliorer la qualité des solutions trouvées. La coévolution de plusieurs agents de recherche permet entre autre une meilleure diversité et ainsi une bonne exploration de l'espace de recherche.
- Evaluation de performances et paysages : L'élaboration de PMO qui sert de benchmarks pour la comparaison des différentes métaheuristiques est nécessaire. Les différents paysages possibles associés à la frontière Pareto doivent être représentés dans les benchmarks. Ceci permet une comparaison des algorithmes par rapport à la structure du paysage du problème. L'extensibilité des algorithmes en terme du nombre d'objectifs a rarement été étudiée. Enfin, les métriques utilisées dans la comparaison doivent être identifiées et communément utilisées.
- Approches d'aide à la décision : La recherche d'un ensemble de solutions Pareto optimales n'est qu'une première étape pour la résolution d'un PMO. La deuxième étape consiste à faire un choix final parmi les solutions trouvées. Pour ce faire, nous pensons que les algorithmes qui ne se basent pas a priori sur des préférences sont généralement plus intéressants, pour les raisons suivantes :
  - Les décideurs désirent en général plusieurs alternatives et non pas une "meilleure" solution unique.
  - La difficulté de déterminer a priori des préférences sans la connaissance du problème.

  Avec les méthodes interactives, les décideurs établissent les préférences de façon progressive, lorsqu'ils apprennent à mieux connaître les caractéristiques du problème. La caractérisation du paysage de la frontière Pareto permet de diriger le décideur vers la solution finale. L'interaction entre les métaheuristiques et le décideur doit donc se baser sur une telle étude.

References

[1] F. Ben Abdelaziz, S. Krichen, and J. Chaouachi. *Meta-heuristics: Advances and trends in local search paradigms for optimization*, chapter A hybrid heuristic for multi-objective knapsack problems, pages 205–212. Kluwer Academic Publishers, 1999.

[2] R. Allenson. Genetic algorithms with gender for multi-function optimisation. Technical Report EPCC-SS92-01, Edinburg Parallel Computing Center, Edinburg, Scotland, 1992.

[3] J. E. Baker. Adaptive selection methods for genetic algorithms. In J. J. Grefenstette, editor, *Int. Conf. on Genetic Algorithms and their Applications*, pages 101–111, Pittsburg, 1985. Lawrence Erlbaum.

[4] D. Beasley, D. R. Bull, and R. R. Martin. A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2):101–125, 1993.

[5] P. J. Bentley and J. P. Wakefield. *Soft Computing in Engineering Design and Manufacturing*, chapter Finding acceptable Pareto-optimal solutions using multiobjective genetic algorithms, pages 231–240. Springer Verlag, London, Jun 1997.

[6] R. L. Carraway, T. L. Morin, and H. Moskowitz. Generalized dynamic programming for multicriteria optimization. *European Journal of Operational Research*, 44:95–104, 1990.

[7] V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making: Theory and Methodology*. North-Holland, New York, 1983.

[8] Y. L. Chen and C. C. Liu. Multiobjective VAR planning using the goal-attainment method. In *IEEE Proc. on Generation, Transmission and Distribution*, volume 141, pages 227–232, May 1994.

[9] C. A. C. Coello. An updated survey of ga-based multiobjective optimization techniques. Technical Report RD-98-08, Laboratorio Nacional de Informtica Avanzada (LANIA), Xalapa, Veracruz, México, Dec 1998.

[10] C. A. C. Coello. Using the min-max method to solve multiobjective optimization problems with genetic algorithms. In *IBERAMIA'98*, LNCS. Springer-Verlag, 1998.

[11] C. A. C. Coello and A. D. Christiansen. An approach to multiobjective optimization using genetic algorithms. In C. H. Dagli et al., editor, *Intelligent Engineering Systems Through Artificial Neural Networks*, volume 5 of *Fuzzy Logic and Evolutionary Programming*, pages 411–416, St. Louis Missouri, USA, Nov 1995. ASME Press.

[12] A. G. Cunha, P. Oliveira, and J. A. Covas. Use of genetic algorithms in multicriteria optimization to solve industrial problems. In T. Back, editor, *Seventh Int. Conf. on Genetic Algorithms ICGA'97*, pages 682–688, San Mateo, California, July 1997. Morgan Kaufmann.

[13] J. Current and M. Marsh. Multiobjective transportation network design and routing problems: Taxonomy and annotation. *European Journal of Operational Research*, 65:4–19, 1993.

[14] P. Czyzak and A. Jaszkiewicz. Pareto simulated annealing - a metaheuristic technique for multiple objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 1998.

[15] G. Dahl, K. Jornsten, and A. Lokketangen. A tabu search approach to the channel minimization problem. In G. Liu, K-H. Phua, J. Ma, J. Xu, F. Gu, and C. He, editors, *Optimization - Techniques and Applications, ICOTA'95*, volume 1, pages 369–377, Chengdu, China, 1995. World Scientific.

[16] K. Deb. Evolutionary algorithms for multi-criterion optimization in engineering design. In *Proc. of Evolutionary Algorithms in Engineering and Computer Science, EUROGEN'99*.

[17] K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. Technical Report CI-49-98, Department of Computer Science, University of Dortmund, Dortmund, Germany, 1998.

[18] K. Deb and D. E. Goldberg. An investigation of niche and species formation in genetic function optimization. In J. D. Schaffer, editor, *Third Int. Conf. on Genetic algorithms ICGA'3*, pages 42–50, San Mateo, California, June 1989. Morgan Kaufmann Pub.

[19] R. P. Dick and N. K. Jha. MOGAC: a multiobjective genetic algorithm for hardware-software cosynthesis of distributed embedded systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(10):920–935, Oct 1998.

[20] M. Ehrgott and K. Klamroth. Non-connected efficiency graphs in multiple criteria combinatorial optimization. In R. Caballero, F. Ruiz, and R. E. Steuer, editors, *Second Int. Conf. on Multi-objective Programming and Goal Programming*, LNCS, pages 140–150, Torremolinos, Spain, May 1996. Springer.

[21] T. A. Ely, W. A. Crossley, and E. A. Williams. Satellite constellation design for zonal coverage using genetic algorithms. Technical Report AAS-98-128, American Astronautical Society, San Diego, USA, Feb 1998.

[22] L. J. Eshelman and J. D. Schaffer. Preventing premature convergence in genetic algorithms by preventing incest. In R. K. Belew and L. B. Booker, editors, *Fourth Int. Conf. on Genetic Algorithms ICGA'4*, pages 115–122, San Mateo, California, 1991. Morgan Kaufmann Pub.

[23] C. M. Fonseca. *Multiobjective genetic algorithms with applications to control engineering problems.* PhD thesis, University of Sheffield, 1995.

[24] C. M. Fonseca and P. J. Fleming. Multiobjective genetic algorithms made easy: selection, sharing and mating restrictions. In *IEEE Int. Conf. on Genetic Algorithms in Engineering Systems: Innovations and Applications*, pages 45–52, Sheffield, UK, 1995.

[25] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[26] M. P. Fourman. Compaction of symbolic layout using genetic algorithms. In J. J. Grefenstette, editor, *Int. Conf. on Genetic Algorithms and their Applications*, pages 141–153, Pittsburgh, 1985.

[27] T. L. Friesz, G. Anandalingam, N. J. Mehta, K. Nam, S. J. Shah, and R. L. Tobin. The multiobjective equilibrium network design problem revisited: A simulated annealing approach. *European Journal of Operational Research*, 65:44–57, 1993.

[28] K. Fujimura. Path planning with multiple objectives. *IEEE Robotics and Automation Society Magazine*, 3(1):33–38, 1996.

[29] K. Fujita, N. Hirokawa, S. Akagi, S. Kimatura, and H. Yokohata. Multi-objective optimal design of automotive engine using genetic algorithm. In *Design Engineering Technical Conferences DETC'98*, pages 1–11, Atlanta, Georgia, Sept 1998.

[30] X. Gandibleux, G. Libert, E. Cartignies, and P. Millot. SMART : Etude de la faisabilité d'un solveur de problèmes de mobilisation de réserve tertiaire. *Revue des systémes de Décision*, 3(1):45–67, 1994.

[31] X. Gandibleux, N. Mezdaoui, and A. Freville. A tabu search procedure to solve multi-objective combinatorial optimization problems. In R. Caballero, F. Ruiz, and R. Steuer, editors, *Second Int. Conf. on Multi-Objective Programming and Goal Programming MOPGP'96*, pages 291–300, Torremolinos, Spain, May 1996. Springer-Verlag.

[32] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, 1989.

[33] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Second Int. Conf. on Genetic Algorithms ICGA'2*, pages 41–49, NJ, 1987. Lawrence Erlbaum.

[34] P. B. Grosso. *Computer simulation of genetic adaptation: Parallel subcomponent interaction in a multilocus model.* PhD thesis, University of Michigan, Microfilms No.8520908 1985.

[35] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[36] D. Halhal, G. A. Walters, D. Ouazar, and D. A. Savic. Water network rehabilitation with a structured messy genetic algorithm. *Journal of Water Resources Planning and Management*, 123(3):137–146, 1997.

[37] M. P. Hansen and A. Jaskiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Technical University of Denmark, March 1998.

[38] A. Hertz, B. Jaumard, C. C. Ribeiro, and W. P. Formosinho Filho. A multi-criteria tabu search approach to cell formation problems in group technology with multiple objectives. *RAIRO Recherche Opérationnelle / Operations Research*, 28(3):303–328, 1994.

[39] J. H. Holland. *Adaptation in natural and artificial systems.* Michigan Press University, Ann Arbor, MI, USA, 1975.

[40] J. Horn. *The nature of niching: Genetic algorithms and the evolution of optimal cooperative populations.* PhD thesis, University of Illinois, Urbana-Champaign, Illinois, 1997.

[41] J. Horn and N. Nafpliotis. Multiobjective optimization using the niched pareto genetic algorithm. Technical Report 93005, University of Illinois, Urbana-Champaign, July 1993.

[42] C. L. Hwang and A. S. M. Masud. Multiple objective decision making - methods and applications. In *Lectures Notes in Economics and Mathematical Systems*, volume 164. Springer-Verlag, Berlin, 1979.

[43] H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 28(3):392–403, Aug 1998.

[44] W. Jakob, M. Gorges-Schleuter, and C. Blume. Application of genetic algorithms to task planning and learning. In R. Manner and B. Manderick, editors, *Parallel Problem Solving from Nature PPSN'2*, LNCS, pages 291–300, Amsterdam, 1992. North-Holland.

[45] B. R. Jones, W. A. Crossley, and A. S. Lyrintzis. Aerodynamic and aeroacoustic optimization of airfoils via a parallel genetic algorithm. In *Proc. of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, number AIAI-98-4811, pages 1–11, 1998.

[46] G. Jones, R. D. Brown, D. E. Clark, P. Willett, and R. C. Glen. Searching databases of two-dimensional and three-dimensional chemical structures using genetic algorithms. In S. Forrest, editor, *Fifth Int. Conf. on Genetic Algorithms*, pages 597–602, San Mateo, California, 1993. Morgan Kaufmann Pub.

[47] F. Kursawe. A variant of evolution strategies for vector optimization. In H. P. Schwefel and R. Manner, editors, *Parallel Problem Solving from Nature*, volume 496 of *Lecture Notes in Computer Science*, pages 193–197, Berlin, 1991. Springer-Verlag.

[48] J. Lis and A. E. Eiben. A multi-sexual genetic algorithm for multi-objective optimization. In T. Fukuda and T. Furuhashi, editors, *Int. Conf. on Genetic Algorithms ICGA*, pages 59–64, Nagoya, Japan, 1996.

[49] X. Liu, D. Begg, and R. J. Fishwick. Genetic approach to optimal topology/controller design of adaptive structures. *International Journal for Numerical Methods in Engineering*, 41:815–830, 1998.

[50] D. H. Loughlin. *Genetic algorithm-based optimization in the development of tropospheric ozone control strategies.* PhD thesis, North Carolina State University, Raleigh, North Carolina, 1998.

[51] D. H. Loughlin and S. Ranjithan. The neighborhood contraint method: A genetic algorithm-based multiobjective optimization technique. In T. Back, editor, *Seventh Int. Conf. on Genetic Algorithms ICGA'97*, pages 666–6773, San Mateo, California, July 1997. Morgan Kaufmann.

[52] S. J. Louis and G. J. E. Rawlins. Pareto optimality, GA-easiness and deception. In S. Forrest, editor, *Fifth Int. Conf. on Genetic Algorithms*, pages 118–123, Univ. of Illinois, Urbana Champaign, 1993. Morgan Kaufmann Publishers.

[53] M-H. Mabed, M. Rahoual, E-G. Talbi, and C. Dhaenens. A genetic algorithm for multicriteria flow shop scheduling. In *15th International Conference on Multiple Criteria Decision Making MCDM'2000*, Ankara, Turkey, Jul 2000.

[54] S. W. Mahfoud. A comparison of parallel and sequential niching methods. In *Sixth Int. Conf. on Genetic Algorithms ICGA'6.*

[55] S. W. Mahfoud. *Niching methods for genetic algorithms.* PhD thesis, University of Illinois, Urbana-Champaign, May 1995.

[56] L. Mandow and E. Millan. Goal programming and heuristic search. In R. Caballero, F. Ruiz, and R. Steuer, editors, *Second Int. Conf. on Multi-Objective Programming and Goal Programming MOPGP'96*, pages 48–56, Torremolinos, Spain, May 1996. Springer-Verlag.

[57] H. Meunier. Métaheuristiques pour l'optimisation multi-critères : Application à la conception de réseaux de radio communication mobiles. Master's thesis, LIFL / University of Lille, Lille, France, July 1999.

[58] H. Meunier, E-G. Talbi, and P. Reininger. A multiobjective genetic algorithm for radio network optimization. In *Congress on Evolutionary Computation CEC'2000*, San Diego, USA, July 2000.
[59] A. Nagar, J. Haddock, and S. Heragu. Multiple and bicriteria scheduling: A literature survey. *European Journal of Operational Research*, 81:88–104, 1995.
[60] S. Obayashi, S. Takahashi, and Y. Takeguchi. Niching and elitist models for multi-objective genetic algorithms. In *Parallel Problem Solving from Nature PPSN'5*, pages 260–269, Amsterdam, Sept 1998. Springer-Verkag.
[61] A. Osyczka and S. Kundu. A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization*, 10(2):94–99, 1995.
[62] V. Pareto. *Cours d'Economie Politique*. Rouge, Lausanne, Switzerland, 1896.
[63] Y. B. Park and C. P. Koelling. An interactive computerized algorithm for multicriteria vehicle routing problems. *Computers and Industrial Engineering*, 16:477–490, 1989.
[64] G. T. Parks and I. Miller. Selective breeding in a multiobjective genetic algorithm. In *Parallel Problem Solving from Nature PPSN'5*, pages 250–259, Amsterdam, Sept 1998. Springer-Verkag.
[65] D. Quagliarella and A. Vicini. *Genetic algorithms and evolution strategies in engineering design*, chapter Coupling genetic algorithms and gradient based optimization techniques, pages 289–309. John Wiley and Sons, Sussex, England, 1997.
[66] B. J. Reardon. Fuzzy logic vs. niched pareto multiobjective genetic algorithm optimization: Part 1: Schaffer's f2 problem. Technical Report LA-UR-97-3675, Los Alamos National Laboratory, New Mexico, Sept 1997.
[67] B. J. Reardon. Fuzzy logic vs. niched pareto multiobjective genetic algorithm optimization: Part 2: A simplified born-mayer problem. Technical Report LA-UR-97-3676, Los Alamos National Laboratory, New Mexico, Sept 1997.
[68] B. J. Reardon. Optimization of micromechanical densification modeling parameters for copper powder using a fuzzy logic based multiobjective genetic algorithm. Technical Report LA-UR-98-0419, Los Alamos National Laboratory, New Mexico, Jan 1998.
[69] J. T. Richardson, M. R. Palmer, G. Liepins, and M. Hilliard. Some guidelines for genetic algorithms with penalty functions. In *Third Int. Conf. on Genetic Algorithms ICGA'3*.
[70] B. J. Ritzel, J. W. Eheart, and S. Ranjithan. Using genetic algorithms to solve a multiple objective groundwater pollution problem. *Water Resources Research*, 30(5):1589–1603, May 1994.
[71] J. Rowe, K. Vinsen, and N. Marvin. Parallel genetic algorithms for multiobjective functions. In J. T. Alander, editor, *Proc. of the Second Nordic Workshop on Genetic Algorithms and Their Applications*, pages 61–70, Vaasa, Finland, Aug 1996.
[72] E. Sandgren. *Advances in design optimization*, chapter Multicriteria design optimization by goal programming, pages 225–265. Chapman and Hall, London, 1994.
[73] S. Sayin and S. Karabati. A bicriteria approach to the two-machine flow shop scheduling problem. *European Journal of Operational Research*, 113:435–449, 1999.
[74] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, *ICGA Int. Conf. on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.
[75] T. Sen, M. E. Raiszadeh, and P. Dileepan. A branch and bound approach to the bicriterion scheduling problem involving total flowtime and range of lateness. *Management Science*, 34(2):254–260, 1988.
[76] P. Serafini. Simulated annealing for multiple objective optimization problems. In *Tenth Int. Conf. on Multiple Criteria Decision Making*, pages 87–96, Taipei, July 1992.
[77] K. J. Shaw and P. J. Fleming. Initial study of multi-objective genetic algorithms for scheduling the production of chilled ready meals. In *Mendel'96 2nd Int. Conf. on Genetic Algorithms*, Brno, Czech Republic, June 1996.
[78] W. M. Spears. Simple subpopulation schemes. In *Third Annual Conf. on Evolutionary Programming*.
[79] N. Srinivas and K. Deb. Multiobjective optimisation using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, 2(8):221–248, 1995.
[80] R. Steuer. *Multiple criteria optimization: Theory, computation and application*. Wiley, New York, 1986.
[81] B. S. Stewart and C. C. White. Multiobjective A*. *Journal of the ACM*, 38(4):775–814, 1991.
[82] P. D. Surry, N. J. Radcliffe, and I. D. Boyd. A multi-objective approach to constraint optimisation of gas supply networks: The COMOGA method. In T. C. Fogarty, editor, *Evolutionary Computing, AISB Workshop*, LNCS, pages 166–180, Sheffield, U.K., 1995. Springer-Verlag.
[83] G. Syswerda and J. Palmucci. The application of genetic algorithms to resource scheduling. In R. K. Belew and L. B. Booker, editors, *Fourth Int. Conf. on Genetic Algorithms ICGA'4*, pages 502–508, San Mateo, California, 1991. Morgan Kaufmann Pub.
[84] E. G. Talbi. *Parallélisme et applications irrégulières*, chapter Algorithmes génétiques parallèles : Techniques et applications, pages 29–48. Hermes, France, 1995.
[85] H. Tamaki, H. Kita, and S. Kobayashi. Multi-objective optimization by genetic algorithms: A review. In *IEEE Int. Conf. on Evolutionary Computation ICEC'96*, pages 517–522, 1996.
[86] V. T'Kindt and J-C. Billaut. Les problèmes d'ordonnancement d'atelier multicritères. Technical Report 206, E3I, Université Francois Rabelais, Tours, France, Sept 1999.
[87] D. S. Todd and P. Sen. A multiple criteria genetic algorithm for container loading. In T. Back, editor, *Seventh Int. Conf. on Genetic Algorithms ICGA'97*, pages 674–681, San Mateo, California, July 1997. Morgan Kaufmann.
[88] D. Tuyttens. An improved MOSA method for solving multi-objective combinatorial optimization problems. submitted, 1998.
[89] E. L. Ulungu. *Optimisation combinatoire multicritère : Détermination de l'ensemble des solutions efficaces et méthodes interactives*. PhD thesis, Université de Mons-Hainaut, 1993.
[90] E. L. Ulungu and J. Teghem. Multi-objective combinatorial optimization problems: A survey. *JMCDA*, 3:83–104, 1994.
[91] E. L. Ulungu and J. Teghem. The two phase method: An efficient procedure to solve bi-objective combinatorial optimization problems. In *Foundations of Computing and Decision Sciences*, volume 20, pages 149–165. 1995.
[92] E. L. Ulungu, J. Teghem, P. Fortemps, and D. Tuyttens. MOSA method: A tool for solving multi-objective combinatorial optimization problems. Technical report, Laboratory of Mathematic and Operational Research, Faculté Polytechnique de Mons, 1998.
[93] D. A. Van Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithm research: A history and analysis. Technical Report TR-98-03, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson, USA, Dec 1998.
[94] D. A. Van Veldhuizen, B. S. Sandlin, R. E. Marmelstein, G. B. Lamont, and A. J. Terzuoli. Finding improved wire-antenna geometries with genetic algorithms. In P. K. Chawdhry, R. Roy, and P. K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, pages 231–240, London, June 1997. Springer Verlag.
[95] M. Visée, J. Teghem, M. Pirlot, and E. L. Ulungu. Two-phases method and branch and bound procedures to solve knapsack problem. *Journal of Global Optimization*, 12:139–155, 1998.
[96] A. Warburton. Approximation of pareto optima in multiple-objective shortest-path problems. *Operations Research*, 35:70–79, 1987.
[97] D. J. White. The set of efficient solutions for multiple-objectives shortest path problems. *Computers and Operations Research*, 9:101–107, 1982.
[98] P. B. Wienke, C. Lucasius, and G. Kateman. Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta*, 265(2):211–225, 1992.

[99] P. B. Wilson and M. D. Macleod. Low implementation cost iir digital filter design using genetic algorithms. In *IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, pages 4/1–4/8, Chelmsford, UK, 1993.

[100] X. Yang and M. Gen. Evolution program for bicriteria transportation problem. In M. Gen and T. Kobayashi, editors, *16th Int. Conf. on Computers and Industrial Engineering*, pages 451–454, Ashikaga, Japan, 1994.

[101] M. Zeleny. *Multiple criteria problem solving.* McGraw-Hill, New York, 1982.

[102] G. Zhou and M. Gen. Genetic algorithm approach on multi-criteria minimum spanning tree problem. *European Journal of Operational Research*, 114:141–152, 1999.

[103] E. Zitzler and L. Thiele. An evolutionary algorithm for multi-objective optimization: The strenght pareto approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology, Zurich, Switzerland, May 1998.

Référence	Approche	Métaheuristique	Problème
[2]	Non Pareto Multi-sexuelle	Algorithme génétique	Construction de pipeline
[1]	Pareto	Hybride séquentiel (RT+AG)	Sac à dos
[12]	Pareto	Algorithme génétique	Extrusion de polymere
[10]	Programmation par but	Algorithme génétique	Design d'une machine à outil
[19]	Pareto	Algorithme génétique	Synthèse de systèmes embarqués distribués
[15]	Agrégation	Recherche Tabou	Affectation de fréquences
[27]	Agrégation	Recuit simulé	Design de réseaux de transport
[29]	Pareto	Algorithme génétique	Design de moteurs d'automobiles
[26]	Non Pareto Sélection lexicographique	Algorithme génétique	Conception de circuits intégrés
[31]	Programmation par but	Recherche Tabou	Permutation
[38]	ϵ-contrainte	Recherche tabou	Formation de cellules en productique
[36]	Pareto	Algorithme génétique (mGA)	Distribution de l'eau
[45]	Non Pareto Sélection parallèle	Algorithme génétique	Design d'airfoil

Référence	Approche	Métaheuristique	Problème
[48]	Non Pareto Multi-sexuelle	Algorithme génétique	Fonctions continues
[51]	ϵ-contrainte	Algorithme génétique	Gestion de la qualité de l'air
[60]	Pareto	Algorithme génétique	Design d'ailes d'avions
[64]	Pareto	Algorithme génétique	Pressurized Water Reactor Reload Design
[66]	Programmation par but	Algorithme génétique Logique flou	Fonctions continues
[67]	Programmation par but	Algorithme génétique Logique flou	Intéraction entre atomes
[68]	Programmation par but	Algorithme génétique Logique flou	Densification micro-mécanique
[71]	Pareto	Algorithme génétique	Fonctions continues
[74]	Non Pareto Sélection parallèle	Algorithme génétique	Fonctions continues
[79]	Pareto	Algorithme génétique	Fonctions continues
[76]	Programmation par but	Recuit Simulé	Voyageur de commerce
[76]	Agrégation	Recuit Simulé	Voyageur de commerce
[77]	Pareto	Algorithme génétique	Ordonnancement
[82]	Non Pareto Sélection parallèle	Algorithme génétique	Design de réseau de gaz

Référence	Approche	Métaheuristique	Problème
[85]	Pareto	Algorithme génétique	Ordonnancement, fonctions continues
[87]	Pareto	Algorithme génétique	Gestion de container
[88]	Agrégation	Hybride séquentiel (Glouton+RS)	Sac à dos
[94]	Agrégation	Algorithme génétique	Design d'antennes
[102]	Hybride (Agrégation+Pareto)	Algorithme génétique	Arbre recouvrant
[103]	Pareto	Algorithme génétique	Sac à dos Fonctions continues