MCours.com

CHAPITRE 3

OPTIMISATION MULTI-OBJECTIFS ET ALGORITHMES ÉVOLUTIONNAIRES

3.1 Introduction

Le chapitre 2 a permis de montrer l'intérêt de résoudre adéquatement le problème d'ordonnancement de voitures dans l'industrie automobile actuelle, mais aussi les difficultés que peuvent entraîner un tel exercice. Même dans sa version théorique, Kis [2004] a prouvé que ce problème appartenait à la classe des problèmes NP-difficiles au sens fort. Le problème industriel s'avère donc davantage complexe en raison des différentes contraintes industrielles qui lui sont spécifiques. Comme la plupart des problèmes d'optimisation rencontrés dans la pratique, le POV industriel est un problème comportant plusieurs objectifs conflictuels. L'optimisation multi-objectifs s'intéresse à la résolution de ce type de problèmes. L'optimisation multi-objectifs n'est pas une science nouvelle, elle est apparue au 19^{ième} siècle avec les travaux en économie de Edgeworth [1881] et de Pareto [1896]. Elle a été appliquée initialement en économie et en gestion avant d'être graduellement utilisée dans d'autres domaines comme l'ingénierie ou l'informatique.

Résoudre un problème d'optimisation consiste souvent à déterminer la ou les meilleures solutions vérifiant un ensemble de contraintes et d'objectifs définis par l'utilisateur. Pour déterminer si une solution est meilleure qu'une autre, il est nécessaire que le problème introduise au moins une mesure de performance permettant d'effectuer une comparaison. Cette mesure de performance correspond souvent à un des objectifs du problème. Ainsi, la meilleure solution, appelée aussi solution optimale, est la solution ayant obtenue la meilleure évaluation par rapport à l'objectif défini. Lorsqu'un seul objectif est spécifié, par exemple un objectif de minimisation de la distance totale, la solution optimale est clairement définie : c'est celle qui a la plus petite distance. Pour les Problèmes Multi-

Objectifs (PMO) le concept de solution optimale devient alors plus difficile à définir. Dans ce cas, la solution recherchée n'est plus un point unique mais un ensemble de solutions dites de compromis aussi appelé ensemble des solutions *Pareto Optimales* (PO). La détermination ou l'approximation de l'ensemble PO n'est qu'une première phase possible dans la résolution pratique d'un PMO, la deuxième phase consistant à utiliser l'expérience du décideur afin d'approfondir la recherche dans une zone plus spécifique de l'ensemble PO.

Ce chapitre introduit, dans un premier temps, les concepts de base de l'optimisation combinatoire multi-objectifs (MOCO). Dans un deuxième temps, il présente les deux types de classification de méthodes de résolution de PMO. Les algorithmes évolutionnaires sont ensuite décrits comme une technique récente d'optimisation qui possède des caractéristiques intéressantes pour résoudre un PMO en portant un intérêt plus particulier sur les algorithmes génétiques. Outre les algorithmes évolutionnaires classiques, nous présentons une autre classe d'algorithmes, moins connus mais aussi inspirés de la biologie, les systèmes immunitaires artificiels. Un large éventail d'algorithmes évolutionnaires multi-objectifs de la littérature est ensuite présenté en essayant d'apporter un regard critique sur chacun d'eux. Finalement, la dernière partie de ce chapitre présente quelques mesures permettant d'évaluer les performances d'un algorithme multi-objectifs

3.2 Optimisation multi-objectifs

La difficulté principale d'un PMO est qu'il n'existe pas de définition de la solution optimale. Le décideur peut simplement exprimer le fait qu'une solution soit préférable à une

autre mais il n'existe généralement pas une unique solution meilleure que toutes les autres. Dès lors, résoudre un PMO ne consiste pas à rechercher la solution optimale mais l'ensemble des solutions satisfaisantes pour lesquelles on ne pourra pas effectuer une opération de classement. Les méthodes de résolution de PMO sont donc des méthodes d'aide à la décision car le choix final est laissé au décideur.

Un PMO peut être défini de la manière suivante dans un contexte de minimisation :

$$PMO \begin{cases} \min \; F(x)=\left(f_1(x),f_2(x),\ldots,f_{nobj}(x)\right) \\ s.c. \; x \in E \end{cases} \qquad (3.1)$$

où $nobj \geq 2$ est le nombre de fonctions objectifs, $x = (x_1,\ldots, x_n)$ est le vecteur représentant les variables de décision, E représente l'ensemble des solutions réalisables associées à des contraintes d'égalité, d'inégalité et des bornes explicites et $F(x) =(f_1(x), f_2(x),\ldots, f_{nobj}(x))$ est le vecteur des objectifs à minimiser.

3.2.1 La notion de dominance et optimum Pareto

Au XIX^{ième} siècle, Vilfredo Pareto, un mathématicien italien, formule le concept suivant : dans un PMO, il existe un équilibre tel que l'on ne peut pas améliorer un objectif sans détériorer au moins un des autres [Pareto 1896].

Cet équilibre a été appelé optimum de Pareto. Un point $x^* \in E$ (ensemble de solutions) est dit *Pareto optimal* s'il n'est dominé par aucun autre point appartenant à E. Autrement dit, un point $x^* \in E$ est Pareto optimal si et seulement s'il n'existe aucun autre point $x \in E$ tel que $f_{obj}(x^*) \leq f_{obj}(x)$ pour tout $obj = 1,\ldots, nobj$ et qu'il existe au moins un $obj2$ tel que

$f_{obj2}(x^*) < f_{obj2}(x)$. Ces points sont également appelés solutions *non inférieures* ou *non dominées*. L'ensemble de tous les points Pareto optimaux constitue la *frontière Pareto*.

Plusieurs autres relations de dominance ont été présentées comme la *a-dominance* [Othomani 1998], la *dominance au sens de Geoffrion* [Ehrgott 2000] ou encore la *cône-dominance* [Collette et Siarry 2002]. Toutefois, dans cette thèse, nous utilisons uniquement la dominance au sens Pareto à partir du concept d'optimum Pareto.

À la Figure 3.1, les solutions représentées par des ronds gris foncés représentent les solutions Pareto optimales. L'ensemble de ces solutions reliées par la ligne en pointillés constitue la frontière Pareto. Les solutions représentées par des ronds blancs représentent l'ensemble des solutions dominées.

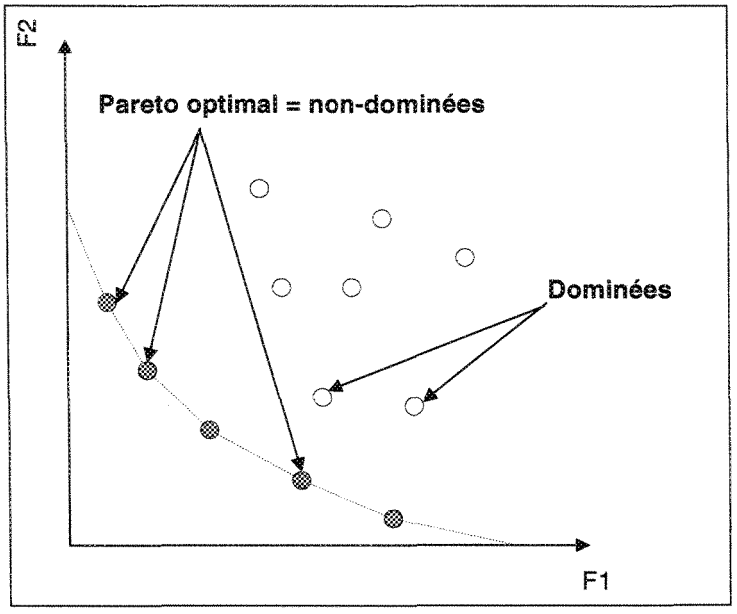

Figure 3.1 : Illustration du concept d'optimum Pareto [Zitzler 2001]

3.2.2 Classification

Dans les différentes publications, nous rencontrons deux classifications différentes des méthodes de résolution de PMO. Le premier classement adopte le point de vue du décideur en classant les méthodes en fonction du moment où l'information sur les préférences du décideur est demandée. Le deuxième classement, plus théorique, trie les méthodes en fonction de leur façon de traiter les fonctions objectifs [Berro 2001].

Dans la *classification décideur*, en fonction du moment où l'information sur les préférences du décideur est demandée, les méthodes de résolution de PMO peuvent être regroupées en trois catégories [Hwang et Masud 1980] : *a priori*, *a posteriori* et *interactive*.

Dans les méthodes *a priori*, l'information sur les préférences du décideur est requise avant la résolution du problème. Dans cette catégorie, les méthodes utilisées pour résoudre les PMO consistent souvent à combiner les différentes fonctions objectif en une fonction d'utilité suivant les préférences du décideur. Dans ce cas, le décideur est supposé connaître *a priori* le poids ou l'importance de chaque objectif afin de les combiner dans une fonction unique. Cela revient à résoudre un problème à objectif unique. Cependant, dans la plupart des cas, le décideur ne peut pas exprimer clairement le poids à attribuer aux différents objectifs, soit par manque d'expérience ou d'informations, soit parce que les différents objectifs sont non commensurables. Des objectifs sont non commensurables si leurs valeurs sont exprimées dans des unités différentes et difficilement comparables.

Dans les méthodes *a posteriori*, la recherche de solutions est effectuée sans qu'aucune information sur les préférences ne soit fournie. Le résultat du processus de recherche est l'ensemble PO ou son approximation à partir duquel le décideur choisira la solution la plus

satisfaisante selon ses préférences. Autrement dit, les compromis sont faits par le décideur après la génération de l'ensemble des solutions non dominées. Si cette approche évite certains inconvénients de l'approche *a priori*, elle exclut l'articulation des préférences par le décideur qui permet de réduire la complexité de l'espace de recherche. Ce type d'approche n'est donc utilisable que dans le cas où la cardinalité de l'ensemble PO est réduite.

Dans les méthodes *interactives*, il y a coopération progressive entre le décideur et le système. Les processus de décision et d'optimisation sont alternés. Par moment, le décideur intervient de manière à spécifier davantage d'informations sur ses préférences. Ces dernières sont alors prises en compte par le système pour la résolution du problème. Le décideur modifie ainsi le compromis entre ses préférences et les résultats. Cette technique permet de faire une exploration guidée de l'ensemble PO. Cependant, il n'est pas garanti que la solution finale soit obtenue après un nombre fini d'itérations.

La *classification concepteur* qui est utilisée dans le reste de la thèse adopte un point de vue plus théorique articulé autour des notions d'agrégation et d'optimum Pareto. Selon cette classification, les approches utilisées pour la résolution de PMO peuvent être regroupées dans les trois catégories suivantes [Talbi 1999] : *les approches basées sur la transformation du problème en un problème uni-objectif*, *les approches non Pareto* et *les approches Pareto*.

L'ensemble des méthodes de la première catégorie repose sur l'axiome suivant : tout décideur essaie inconsciemment de maximiser une fonction d'utilité U [Berro 2001] :

$$U = U(f_1, f_2, \ldots, f_{nobj}). \qquad (3.2)$$

Dans cette catégorie d'approches, les modèles les plus utilisés sont le modèle additif :

$$U = \sum_{obj=1}^{nobj} U_{obj}(f_{obj}) \qquad (3.3)$$

et le modèle multiplicatif :

$$U = \prod_{obj=1}^{nobj} U_{obj}(f_{obj}) \qquad (3.4)$$

Cependant, l'utilisation de ces modèles impose généralement que les objectifs soient commensurables. Il est donc très difficile d'utiliser ces techniques lorsque l'ensemble des objectifs est composé à la fois d'objectifs qualitatifs et quantitatifs. Parmi les techniques représentatives de cette classe, il y a la *moyenne pondérée* et la *programmation par but.*

La technique de la moyenne pondérée consiste à additionner tous les objectifs en affectant à chacun d'eux un coefficient de poids. Ce coefficient représente l'importance relative que le décideur attribue à l'objectif. Cela modifie un PMO en un problème uni-objectif de la forme :

$$\min \sum_{obj=1}^{nobj} w_{obj} f_{obj}(x) \qquad (3.5)$$

où w_{obj} représente le poids affecté à l'objectif *obj*, $w_{obj} \geq 0$ et $\sum_{obj=1}^{nobj} w_{obj} = 1$. Si ce genre de méthode est facile à mettre en œuvre, la détermination des poids s'avère être une question délicate qui détermine l'efficacité de la méthode. De plus, cette méthode est généralement sensible à la forme de la frontière Pareto [Francisci 2002].

Dans la programmation par but, le décideur fixe un but G_{obj} à atteindre pour chaque objectif *obj* [Charnes et Cooper 1961]. Ces valeurs sont ensuite ajoutées au problème comme des contraintes supplémentaires. La nouvelle fonction objectif est alors modifiée de façon à minimiser la somme des écarts entre les résultats et les buts à atteindre :

$$\min \sum_{obj=1}^{nobj} | f_{obj}(x) - G_{obj} | \qquad (3.6)$$

Différentes variantes et applications de cette technique ont été proposées [Ignizio 1981; Van Veldhuizen 1999]. Comme pour la somme pondérée, cette méthode est facile à mettre en œuvre. De plus, elle fournit un résultat même si un mauvais choix initial a conduit le décideur à donner un ou plusieurs buts T_i non réalisables. Cependant, l'efficacité de la méthode dépend fortement de la définition des buts à atteindre [Berro 2001].

Les *approches non Pareto*, de leur côté, ne transforment pas le PMO en un problème uni-objectif et n'utilisent pas les concepts de dominance ou d'optimum Pareto. En général, cette classe de méthodes possède un processus de recherche qui traite séparément les différents objectifs. Dans cette catégorie, une des techniques les plus répandues est la *sélection lexicographique*.

La sélection lexicographique a été introduite par Fourman [1985]. Dans cette technique, les objectifs sont préalablement rangés par ordre d'importance par le décideur. Ensuite, l'optimum est obtenu en minimisant tout d'abord la fonction objectif la plus importante puis la deuxième, et ainsi de suite sans jamais détériorer les objectifs précédents. Le risque essentiel de cette méthode est la grande importance attribuée aux objectifs classés en premier. La meilleure solution f_1* trouvée pour l'objectif le plus important risque de faire

converger l'algorithme vers une zone restreinte de l'espace des solutions et enfermer les points dans cette région [Berro 2001].

La dernière catégorie, les *approches Pareto,* utilise directement la notion d'optimum Pareto dans leur processus de recherche. Le processus de sélection des solutions générées est basé sur la notion de non-dominance. Le principal avantage de ces méthodes est qu'elles peuvent générer des solutions PO dans toutes les régions de la frontière Pareto. Certaines techniques de résolution appartenant à cette catégorie sont présentées plus en détail dans la Section 3.6 du présent chapitre.

Classiquement, les PMO sont résolus à l'aide de techniques comme la moyenne pondérée ou la programmation par but car ces techniques permettent souvent d'utiliser des algorithmes éprouvés pour des problèmes uni-objectif [Francisci 2002]. Cependant, Deb [1999] a montré que certaines de ces techniques avaient un champ d'applications limité. De plus, les méthodes classiques ont toutes en commun qu'elles requièrent généralement plusieurs étapes d'optimisation pour obtenir une approximation de l'ensemble PO [Francisci 2002].

Récemment, les algorithmes évolutionnaires se sont révélés être une alternative intéressante aux méthodes classiques grâce à leur faculté à exploiter de vastes espaces de recherche et à générer des compromis multiples en une seule étape d'optimisation [Francisci 2002]. De plus, les algorithmes évolutionnaires sont peu sensibles à la forme de la frontière Pareto. Les algorithmes évolutionnaires font partie des méthodes métaheuristiques. Ce sont des techniques d'approximation conçues dans le but de s'attaquer à des problèmes d'optimisation complexes qui n'ont pu être résolus de façon efficace par

les heuristiques et les méthodes d'optimisation classique [Osman et Laporte 1996]. Parmi les métaheuristiques les plus utilisées, citons le recuit simulé [Kirkpatrick *et al.* 1983], la recherche avec tabous [Glover 1986], l'optimisation par colonie de fourmis [Dorigo 1992] et bien entendu les algorithmes génétiques [Holland 1975] qui font partie de la famille des algorithmes évolutionnaires dont les principes de base sont brièvement décrits à la section suivante.

3.3 Les Algorithmes Évolutionnaires (AE)

Les algorithmes évolutionnaires sont des méthodes qui tentent de simuler le processus de l'évolution naturelle dans un environnement hostile lié au problème à résoudre. Schématiquement, on peut les assimiler à un processus d'optimisation où des individus évoluent dans le temps, afin de devenir de plus en plus adapté à un environnement donné : le problème à résoudre. Les AE se sont avérés être des alternatives efficaces aux méthodes classiques pour la résolution de nombreux problèmes d'optimisation uni-objectifs et multi-objectifs. En général, les algorithmes évolutionnaires sont divisés en quatre grandes classes :

- les algorithmes génétiques [Holland 1975; Goldberg 1989];
- les stratégies d'évolution [Rechenberg 1965; Schewfel 1981];
- la programmation évolutive [Fogel *et al.* 1966]; et
- la programmation génétique [Koza 1992].

Cette section présente brièvement les quatre classes d'algorithmes évolutionnaires en insistant plus particulièrement sur les algorithmes génétiques.

3.3.1 Les Algorithmes Génétiques (AG)

Les algorithmes génétiques tentent de simuler le processus de sélection naturelle dans un environnement hostile lié au problème à résoudre en s'inspirant des théories de l'évolution proposées par Charles Darwin. Les AG sont à la base des algorithmes d'optimisation stochastiques, mais ils peuvent également servir pour l'apprentissage automatique. Les premiers travaux dans ce domaine ont commencé dans les années cinquante lorsque plusieurs biologistes américains ont simulé des structures biologiques sur ordinateur. Entre 1960 et 1970, John Holland, sur la base des travaux précédents, développe les principes fondamentaux des algorithmes génétiques dans le cadre de l'optimisation mathématique [Holland 1975]. Malheureusement, les ordinateurs de l'époque n'étaient pas assez puissants pour envisager l'utilisation des AG sur des problèmes réels de grande taille. La parution en 1989 de l'ouvrage de référence écrit par Goldberg qui décrit l'utilisation de ces algorithmes dans le cadre de résolution de problèmes concrets a permis de mieux faire connaître ces derniers dans la communauté scientifique et a marqué le début d'un nouvel intérêt pour cette technique d'optimisation qui demeure néanmoins très récente. La Figure 3.2 illustre le fonctionnement général d'un algorithme génétique standard.

Générer aléatoirement une population *Pop* de |*Pop*| individus
TANT QUE Nombre de générations ≤ Nombre de générations Maximum **FAIRE**
Évaluation : Calculer la fitness de chaque individu x
Sélection : Établir de façon probabiliste les paires d'individus qui vont se reproduire en accordant une plus grande chance aux meilleurs individus
Croisement : Recombiner les paires sélectionnées en fonction de p_c .
Mutation : Effectuer une mutation en fonction de p_m
Remplacement : Les nouveaux individus produits représentent la nouvelle population

Figure 3.2 : Un algorithme génétique standard

Le but principal d'un AG est de chercher, dans un espace de solutions E, un élément de cet espace ayant la meilleure adaptation à l'environnement posé. Chaque élément de E est un individu, noté x. Une population *Pop* est donc un ensemble de $|Pop|$ éléments (individus) de E : $Pop = (x_1, x_2, \ldots, x_{|Pop|})$. La mesure du degré d'adaptation de chaque individu constitue son indice de qualité f (*fitness*). L'objectif d'un AG est de faire évoluer cette population *Pop* afin de trouver le meilleur individu x^*. Dans ce but, à chaque génération t, les individus de la population $Pop(t)$ sont sélectionnés, croisés et mutés selon des probabilités prédéfinies respectivement p_c (probabilité de croisement) et p_m (probabilité de mutation).

Bien que les principes sous-jacents soient simples, ces algorithmes s'avèrent être des mécanismes de recherche généraux, puissants et robustes [Francisci 2002]. De plus, les AG semblent être spécialement utiles en optimisation multi-objectifs car ils sont capables de déterminer plusieurs solutions en une seule exécution. Certains auteurs suggèrent même que l'optimisation multi-objectifs peut être un domaine où les AG font mieux que d'autres méthodes de recherche heuristique [Fonseca et Fleming 1995; Valenzuela-Rendon et Uresti-Charre 1997].

La suite de cette section présente sommairement le fonctionnement d'un AG. Cette présentation permettra d'introduire certaines notions qui seront utiles pour la compréhension du reste de ce document.

3.3.1.1 Représentation des individus

Dans la nature, les structures géniques sont codées en base 4, dont les « chiffres » sont les quatre bases azotées : l'*adénine*, la *thymine*, la *cytosine* et la *guanine*. Dans le cadre des

AG, ce type de codage est bien difficile à utiliser et n'a donc pas été retenu. Nous présentons ici deux des représentations utilisées pour le codage des individus dans un AG à savoir la représentation *binaire naturelle* et la représentation *par chemin*.

Historiquement, la représentation binaire est la première représentation utilisée par les AG. Chaque individu x est représenté sous forme de chaînes de bits où chaque élément prend la valeur 0 ou 1 :

$$x = (e_1, e_2, \ldots, e_L) \in \{0,1\}^L, \qquad (3.7)$$

Où L est la taille du vecteur (nombre de bits). L'espace de recherche est alors $\{0,1\}^L$. Ce type de représentation possède plusieurs avantages : alphabet minimum, facilité de mise en place d'opérateurs génétiques et existence de résultats théoriques [Goldberg 1989]. Néanmoins, cette représentation présente deux inconvénients majeurs :

- les performances de l'algorithme sont diminuées lorsque la longueur de la chaîne augmente; et
- deux éléments voisins en terme de distance de Hamming (représente le nombre de bits dont diffèrent deux nombres binaires) ne codent pas nécessairement deux éléments proches dans l'espace de recherche.

La représentation par chemin est une manière naturelle de coder une solution d'un Problème de Voyageur de Commerce (PVC) [Potvin 1996]. Chaque individu x est représenté sous forme de chaînes de symboles. Deux symboles successifs d'un individu x représentent deux numéros de villes adjacentes dans le chemin proposé. Pour être valide, un individu doit comporter chaque symbole (chaque ville à visiter dans le cas d'un PVC) et ce, une et une seule fois.

3.3.1.2 La sélection

La sélection est un des principaux opérateurs utilisés par les AG. Son rôle est de sélectionner les individus relativement performants afin qu'ils se reproduisent. Plusieurs stratégies de sélection ont été mises en place et les paragraphes suivants décrivent deux des plus répandues : *la roulette* et *le tournoi.*

La première sélection mise en place pour les AG est le *tirage à la roulette* (*Roulette Wheel Selection* (RWS)) introduite par Goldberg [1989]. C'est une méthode stochastique qui exploite la métaphore d'une roulette de casino. Elle consiste à associer à chaque individu un segment (ou case de la roue) dont la longueur est proportionnelle à sa fitness comme l'illustre la Figure 3.3. La roue étant lancée, l'individu sélectionné est celui sur lequel la roue s'est arrêtée. Cette méthode favorise les meilleurs individus, mais tous les individus conservent néanmoins des chances d'être sélectionnés.

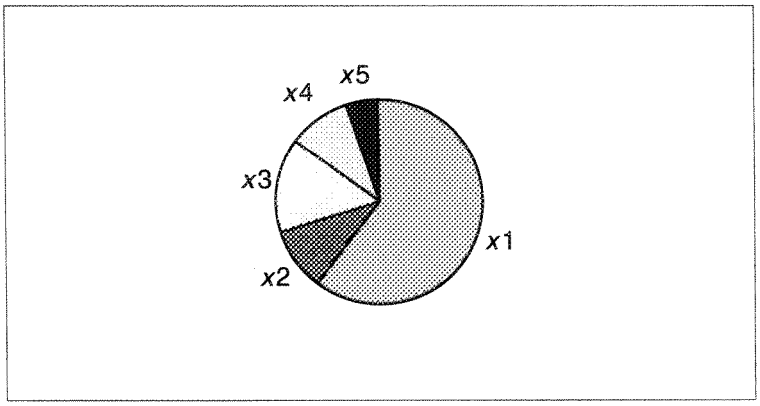

Figure 3.3 : Roulette pour une population de 5 individus avec $f(x_i) = \{60,10, 15,10, 5\}$.

De son côté, *la sélection par tournoi* fait intervenir le concept de comparaison entre les individus. Son principe est de choisir un groupe de q individus aléatoirement dans la

population et de sélectionner d'une manière déterministe le meilleur dans ce groupe. Au cours d'une génération, il y a autant de tournois que d'individus à sélectionner. La pression de sélection est ajustée par le nombre de participants q à un tournoi. Un q élevé a une forte pression de sélection et inversement. L'avantage de cette technique de sélection est qu'elle n'est pas coûteuse à mettre en œuvre et à exécuter.

3.3.1.3 Le croisement

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants mais il est tout à fait possible d'imaginer des croisements avec α parents pour produire β enfants. Un croisement consiste à échanger les gènes des parents afin de donner des enfants qui comportent des propriétés combinées. Bien qu'il soit souvent aléatoire, cet échange d'informations offre aux algorithmes génétiques une part de leur puissance : quelquefois, de *bons* gènes d'un parent viennent remplacer les *mauvais* gènes d'un autre et créent des enfants mieux adaptés que les parents. Dans le paragraphe suivant, nous allons présenter le *croisement à un point* qui est une des premières stratégies de croisement utilisées par les AG [Holland 1962].

Le but de cette stratégie de croisement est d'échanger un fragment de gènes entre deux individus. Le croisement à un point consiste à choisir au hasard un point de croisement pour chaque couple d'individus, et ensuite à échanger un fragment de gènes entre les deux individus afin de créer deux enfants. À la Figure 3.4, le point de coupure entre les deux parents se situe à la position 5. À partir du premier parent, le premier enfant est obtenu en copiant les gènes du deuxième parent à droite du point de coupure. Le deuxième enfant est

obtenu à partir du deuxième parent en copiant les gènes du premier parent à droite du point de coupure.

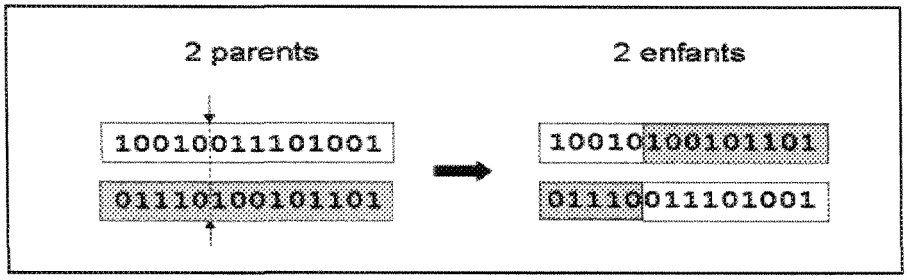

Figure 3.4 : Illustration du croisement à un point

3.3.1.4 La mutation

L'opérateur de mutation joue le rôle de bruit et tente d'empêcher une convergence trop hâtive. Plusieurs opérateurs de mutation ont été mis en place, parmi lesquels l'inversion et l'échange. La mutation par inversion est souvent utilisée avec la représentation binaire [Ben Hamida 2001]. Elle consiste simplement à inverser de manière aléatoire un gène d'un individu. À la Figure 3.5, la position 9 est sélectionnée et le gène est inversé, il passe de 1 à 0.

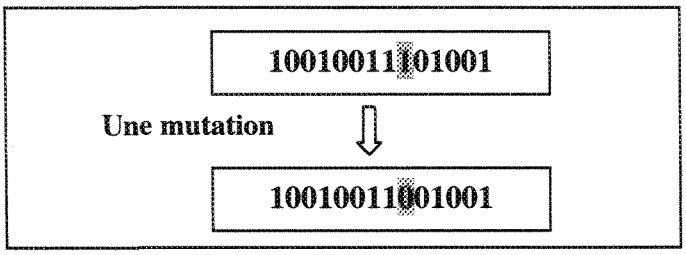

Figure 3.5 : Illustration de la mutation par inversion

La mutation par échange consiste à sélectionner de manière aléatoire deux gènes d'un individu et d'échanger les positions respectives des deux éléments choisis. À la Figure 3.6, les gènes aux positions 4 et 6 sont échangés.

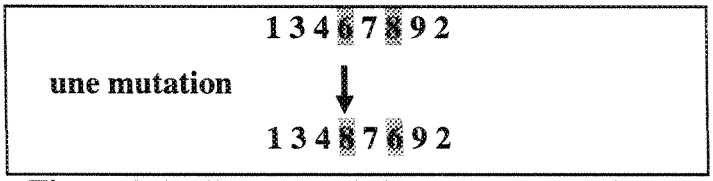

Figure 3.6 : Illustration de la mutation par échange

3.3.2 Les stratégies d'évolution

Les stratégies d'évolution constituent une technique d'AE développée par Rechenberg et Schwefel au milieu des années 1960. Ces techniques représentent chaque individu comme un ensemble de caractéristiques de la solution potentielle. En général, cet ensemble prend la forme d'un vecteur de nombres réels de dimension fixe. Les stratégies d'évolution utilisent une population de parents, de taille μ, à partir de laquelle des individus sont sélectionnés aléatoirement afin de générer une population d'enfants, de taille $\lambda \geq \mu$. Ceux-ci sont ensuite modifiés à l'aide des mutations qui consistent simplement à ajouter une valeur générée aléatoirement selon une fonction de densité de probabilité. Les opérateurs de mutation utilisé dans les stratégies d'évolution sont souvent les mêmes que ceux définis pour les algorithmes génétiques comme par exemple la mutation par échange ou la mutation par inversion présentées à la section précédente. Les paramètres de la fonction de densité de probabilité, nommés les *paramètres de la stratégie*, évoluent eux aussi dans le temps selon les mêmes principes que les paramètres caractérisant les individus. Pour former la nouvelle population parents, deux stratégies peuvent être employées :

- la première stratégie, l'approche (μ, λ), consiste à choisir les μ meilleurs individus parmi les λ enfants;

- la deuxième technique, l'approche (μ + λ), consiste à retenir les μ meilleurs individus dans les populations parents et enfants combinées.

3.3.3 La programmation évolutive

La programmation évolutive a été initialement conçue pour des machines à états finis et a été par la suite étendue aux problèmes d'optimisation [Fogel *et al.* 1966]. Cette approche met l'emphase sur la relation entre les parents et leurs descendants plutôt que de simuler des opérateurs génétiques d'inspiration naturelle. Contrairement aux autres AE classiques, la programmation évolutive n'utilise pas une représentation spécifique mais plutôt un *modèle évolutionnaire* jumelé à une représentation des solutions et à un opérateur de mutation choisi en fonction du problème à résoudre.

Pour effectuer la programmation évolutive, une population de μ solutions potentielles au problème est d'abord générée de façon aléatoire. Chaque individu x de la population ainsi produite génère par la suite λ enfants par mutation. Une opération de sélection naturelle est alors appliquée afin de former une nouvelle population de μ individus. Le processus de mutation/sélection est répété de manière itérative jusqu'à ce qu'une solution acceptable soit trouvée.

3.3.4 La programmation génétique

La programmation génétique [Koza 1992] est un paradigme permettant la programmation automatique d'ordinateurs par des heuristiques basées sur les mêmes principes d'évolution que les AG à savoir : sélection, croisement et mutation. La différence principale entre ces deux AE se situe au niveau de la représentation des individus. En effet,

dans la programmation génétique on fait évoluer des individus dont la structure est similaire à celle de programmes informatiques.

Initialement, la programmation génétique représente les individus sous forme d'arbres, c'est-à-dire des graphes orientés et sans cycle, dans lesquels chaque nœud est associé à une opération élémentaire relative au domaine du problème. Plusieurs autres représentations comme des programmes linéaires [Banzhaf *et al.* 1998] et des graphes cycliques [Teller et Veloso 1996], ont été par la suite utilisées. La programmation génétique s'avère particulièrement adaptée pour faire évoluer des structures complexes de dimension variable.

3.3.5 AE et optimisation multi-objectifs

C'est dans le milieu des années 80 [Schaffer 1985] que, pour la première fois, les AE ont été utilisés pour l'optimisation multi-objectifs. Depuis, plusieurs implémentations différentes des AE ont été proposées et appliquées avec succès à divers PMO [Hajela et Lin 1992; Fonseca et Fleming 1993; Horn *et al.* 1994; Srivinas et Deb 1994; Ishibuchi et Murata 1996; Valenzuela-Rendon et Uresti-Charre 1997]. Récemment, certains chercheurs ont étudié quelques aspects des AE pour l'optimisation multi-objectifs comme la convergence vers la frontière Pareto [Rudolph 1998] ou l'*élitisme* [Obayashi *et al.* 1998; Parks et Miller 1998]. Parallèlement, de nouvelles techniques qui exploitent la *notion de dominance* au sens Pareto ont été développées [Zitzler et Thiele 1998; Knowles et Corne 2000; Coello Coello et Pulido 2001].

3.3.5.1 Maintenir la diversité : les techniques de nichage

Une des principales difficultés à surmonter lorsque l'on veut utiliser un AE et en particulier un AG, pour l'optimisation multi-objectifs est le maintien de la diversité. En effet, un algorithme génétique simple a, en général, tendance à converger vers une solution unique en négligeant souvent les autres solutions [Mahfoud 1995]. Dans un contexte multi-objectifs, on ne cherche pas une mais un ensemble de solutions. Il est donc essentiel de préserver une certaine diversité entre les différentes solutions trouvées à chaque itération par un AG multi-objectifs. Les techniques de nichage ont été développées en ce sens. Ces techniques ont pour objectif la formation et le maintien de sous-populations stables appelées *niches* dans une même population. Les paragraphes suivants décrivent deux des principales techniques de nichage : le *partage de fitness* et la *méthode de remplacement.*

Le *partage de fitness* est une technique introduite par Goldberg et Richardson [1987]. Dans cette méthode, la fitness d'un individu est dégradée en fonction du nombre d'individus qui lui sont proches dans la population. Le but principal de cette méthode est de pénaliser les individus qui sont trop proches les uns des autres en terme de distance. Ainsi, la fitness partagée f' d'un individu x est donnée par :

$$f'(x) = \frac{f(x)}{\sum_{\forall\, y \in Pop} sh\,(dist\ \ (x, y))} \qquad (3.8)$$

où *sh* désigne la fonction de partage qui a comme paramètre d'entrée la distance *(dist)* entre deux solutions. Elle retourne 1 si les deux solutions sont identiques, 0 si la distance entre les deux dépasse un certain seuil (σ_{sh}). La somme des valeurs des fonctions de partage est

appelée *compte de niche*. Une des fonctions de partage les plus utilisées est celle proposée par Goldberg et Richardson [1987] :

$$sh(dis)=\begin{cases} 1-\left(\dfrac{dis}{\sigma_{sh}}\right)^{\alpha} & si\ dis(x,y)<\sigma_{sh} \\ 0 & \text{sinon} \end{cases} \qquad (3.9)$$

où α est une constante de contrôle de la fonction de partage et σ_{sh} représente le rayon de la niche. Le calcul de la distance *(dist)* peut se faire à l'aide d'une distance de *Hamming* ou d'une distance *Euclidienne*.

La *méthode de remplacement* a été introduite par De Jong [1975]. Le principe originel consiste à remplacer des éléments d'une population par d'autres éléments similaires et meilleurs qu'eux. Cette méthode est une analogie des systèmes vivants en compétition pour une ressource. Dans les problèmes d'optimisation, la ressource est l'optimum à atteindre. Des individus situés dans des zones différentes de l'espace des solutions ne sont pas en concurrence pour le même optimum, alors que des individus proches sont en compétition. En remplaçant les individus semblables, la méthode de remplacement permet de conserver les différentes niches de la population tout en introduisant de la diversité dans la population. Cependant, Mafhoud [1995] a prouvé que bien qu'une telle méthode permette d'obtenir une certaine diversité dans la population, elle s'avère peu efficace pour bien approximer la frontière Pareto. Ceci explique, en partie, pourquoi cette technique est moins utilisée dans les AE que le partage de fitness [Blickle et Thiele 1996].

3.4 Les Systèmes Immunitaires Artificiels (SIA)

Le terme système immunitaire artificiel (*Artificial Immune System*) s'applique, en général, aux algorithmes inspirés du fonctionnement du système immunitaire des organismes vertébrés. Même si cette classe d'algorithmes est moins connus et répandus que les AE, un nombre croissant de systèmes immunitaires artificiels ont été proposés ces dernières années pour être appliqués dans différents domaines, tels que la robotique, la classification, la détection d'anomalies ou encore l'optimisation [De Castro et Von Zuben 2000].

D'un point de vue biologique, le système immunitaire est responsable de la protection de l'organisme contre les agressions extérieures. Pour cela, les cellules de l'organisme possèdent sur leur membrane un certain nombre de molécules spécifiques appelées *antigènes*. L'ensemble des antigènes présents sur les cellules constituent un identifiant unique pour chaque organisme. Dans le système immunitaire des organismes vertébrés, les *lymphocytes* sont des cellules qui possèdent des récepteurs capables de se lier spécifiquement à un antigène donné, permettant ainsi de reconnaître une cellule étrangère à l'organisme. Lorsqu'un lymphocyte reconnaît une cellule étrangère, il va être stimulé puis proliférer avant de se différencier en cellules permettant de reconnaître l'antigène ciblé, ou en cellule permettant de combattre les agressions. Dans le premier cas, l'organisme sera capable de réagir plus rapidement à une nouvelle exposition à l'antigène, c'est le principe de des vaccins. Dans le second cas, l'organisme combattra les agressions en secrétant des *anticorps*. Ces derniers sont en fait des molécules réceptrices permettant de reconnaître l'antigène et de le bloquer. Le processus de prolifération des lymphocytes se fait par la

production de *clones*, tandis que la différentiation, quant à elle, se fait par un mécanisme *hyper-mutation* des cellules clonées. C'est cette métaphore qui est utilisée dans les SIA, en particulier le processus de sélection par clonage et de rétroaction permettant la multiplication et la mémoire du système. Certains auteurs considèrent l'approche utilisée dans les SIA très proche de celle d'un AE ou d'un réseau de neurones [Dréo et Siarry 2003; Coello Coello et Cortés 2005]. L'algorithme 3.1 résume le fonctionnement général d'un SIA simple.

Algorithme 3.1 : un SIA standard [Dréo et Siarry 2003]

1: Générer un ensemble de solutions POP, composé d'un ensemble de cellules mémoires POP_M ajoutées à la population courante POP_t : $POP = POP_M + POP_t$
2 : Déterminer les n meilleures cellules de POP en utilisant un critère d'affinité
3 : Cloner les n individus sélectionnés pour former une population de clones C_{pop}. Le nombre de clones produits pour chaque individu est fonction de l'affinité;
4 : Effectuer une hyper mutation des clones produit afin d'obtenir une population de clones mutés C^*_{POP}. La mutation est proportionnelle à l'affinité ;
5 : Sélectionner les individus de C^*_{POP} pour former la nouvelle population mémoire POP_M;
6 : Éliminer les plus mauvais individus de POP pour former POP_{t+1} ;
7 : Si aucun critère d'arrêt n'est pas atteint, retourner en 1

En se basant sur l'Algorithme 3.1, on constate qu'on peut, dans le cadre de l'optimisation combinatoire, considérer les SIA comme une forme d'algorithme évolutionnaire utilisant des opérateurs particuliers. Par exemple, pour effectuer la sélection, on se fonde principalement sur une mesure d'affinité entre le récepteur d'un anticorps et d'un antigène; la mutation s'opère pour sa part via un opérateur d'hyper mutation issu de la métaphore immune [Dréo et Siarry 2003]. Une description détaillée des fondements théoriques des SIA peut être trouvée dans [Dasgupta et Attoh-Okine 1997; De Castro et Von Zuben 1999; De Castro et Von Zuben 2000]. Les SIA ont été surtout appliqués dans

les domaines de l'extraction de connaissances [Timmis et Knight 2001], de l'extraction de règles [Carvalho et Freitas 2001], du clustering [De Castro et Von Zuben 2000; Nasraoui *et al.* 2002], de détection de fraudes [Lee *et al.* 2000; Overill 2002] ou encore de classification [Carter 2000; Twycross et Cayzer 2002]. On retrouve aussi des SIA pour l'optimisation multimodale [Forrest et Perelson 1991; Smith *et al.* 1992; Smith *et al.* 1993; de Castro et Von Zuben 2002] et l'optimisation uni-objectif [de Castro et Von Zuben 2002]. Toutefois, malgré certaines similitudes entre les AE et les SIA, on note que contrairement aux AE, les SIA ont très peu été appliqués pour l'optimisation multi-objectifs [de Castro et Von Zuben 2002; Coello Coello et Cortés 2005].

La section suivante présente un éventail des méthodes de résolution de PMO basées sur les AE en les regroupant selon la *classification concepteur* décrite à la Section 3.2.2.

3.5 AE basés sur la transformation du problème en un problème uni-objectif

Les implémentations des AG de cette catégorie d'approche sont basées sur les techniques classiques pour générer des compromis sur l'ensemble PO. Certaines approches comme le *Weight-Based Genetic Algorithm* (*WBGA*) [Hajela et Lin 1992] ou le *Multi-Objective Genetic Local Search* (*MOGLS*) [Ishibuchi et Murata 1996] utilisent la technique de la moyenne pondérée. Comme chaque individu utilise une combinaison particulière de poids qui est soit choisie aléatoirement, soit encodée dans l'individu, tous les membres de la population sont évalués par différentes fonctions objectifs. En conséquence, l'optimisation est effectuée dans plusieurs directions simultanément [Francisci 2002].

Cependant, les inconvénients potentiels des techniques classiques, comme la sensibilité à la forme de la frontière Pareto pour certaines, peuvent restreindre l'efficacité de ce type d'AG [Van Veldhuizen 1999].

3.6 AE non Pareto

En général, les méthodes dites non Pareto possèdent un processus de recherche qui traite séparément les objectifs. Cette section présente un algorithme génétique représentatif de cette classe d'approche : le *Vector Evaluated Genetic Algorithm* (VEGA).

En 1985, Schaffer propose une extension d'un algorithme génétique simple pour la résolution d'un PMO [Schaffer 1985]. Cette méthode est appelée *Vector Evaluated Genetic Algorithm*. La seule différence avec un algorithme génétique simple est la manière dont s'effectue la sélection. Comme le montre la Figure 3.7, l'idée est simple. Pour un problème comportant *nobj* objectifs et une population de $|POP|$ individus, une sélection de $|POP|/nobj$ individus est effectuée pour chaque objectif. Ainsi, *nobj* sous-populations vont être créées, chacune d'entre elles contenant les $|POP|/nobj$ meilleurs individus pour un objectif particulier. Les *nobj* sous-populations sont ensuite mélangées afin d'obtenir une nouvelle population de taille $|POP|$. Le processus de sélection se termine par l'application des opérateurs génétiques de croisement et de mutation.

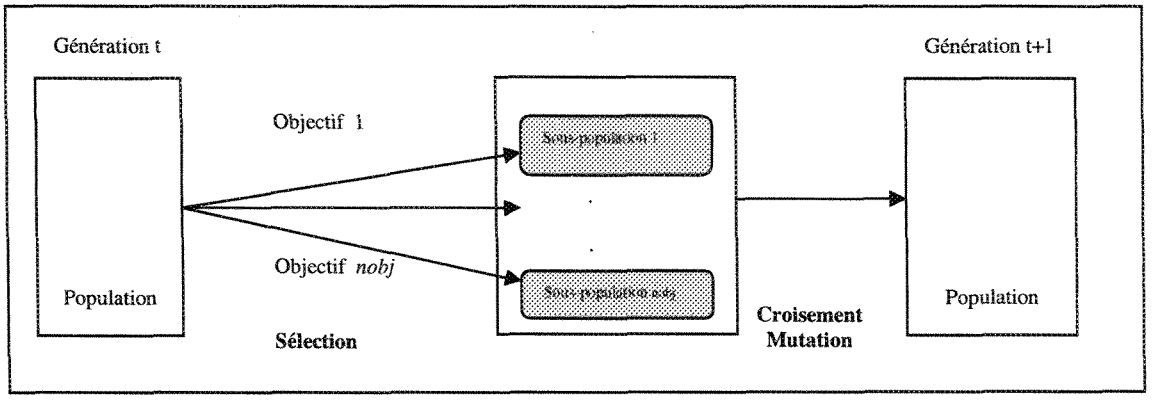

Figure 3.7 : Sélection parallèle dans l'algorithme VEGA [Talbi 1999]

Le VEGA est l'un des premiers AG multi-objectifs. Cette technique est très facilement implémentable dans un AG classique et ceci explique sans doute pourquoi elle est encore très souvent utilisée [Berro 2001]. Cependant, avec une telle sélection, les points se répartissent sur les extrêmes de la frontière Pareto au détriment des solutions de *compromis* qui ont une fitness générale acceptable mais ne possèdent aucun objectif fort [Talbi 1999].

3.7 AE Pareto

L'idée d'utiliser la dominance au sens de Pareto dans un AE a été proposée par Goldberg [1989] pour résoudre les problèmes proposés par Schaffer [1985]. L'utilisation d'une sélection basée sur la notion de dominance de Pareto va faire converger la population vers un ensemble de solutions efficaces. Ce concept ne permet pas de choisir une alternative plutôt qu'une autre, mais il apporte une aide précieuse au décideur.

Les paragraphes suivants présentent des techniques inspirées des algorithmes évolutionnaires qui utilisent cette notion. Cette présentation est divisée en deux parties : dans un premier temps, nous allons nous intéresser aux techniques non élitistes et, dans un deuxième temps, nous aborderons les techniques élitistes.

3.7.2 Les techniques non élitistes

Une méthode de résolution est dite non élitiste lorsqu'elle ne comporte aucun mécanisme explicite permettant la conservation des meilleures solutions tout au long de son exécution.

3.7.2.1 Multiple Objective Genetic Algorithm (MOGA)

Proposé par Fonseca et Fleming [1993], le MOGA est une méthode dans laquelle chaque individu de la population est rangé en fonction du nombre d'individus qui le domine. Ensuite, l'algorithme utilise une fonction de calcul de la fitness permettant de prendre en compte le rang de l'individu et le nombre d'individus ayant le même rang.

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

$$\text{rang}(x, t) = 1 + \alpha(t) \qquad (3.10)$$

La procédure de sélection utilise ensuite ces rangs pour sélectionner ou éliminer des blocs d'individus. De plus, les auteurs calculent la fitness des individus de la façon suivante :

1. calcul du rang de chaque individu; et
2. affectation de la fitness de chaque individu par application d'une fonction de changement d'échelle sur la valeur de son rang. Cette fonction est en général linéaire. Suivant le problème, d'autres types de fonction pourront cependant être

envisagés afin d'augmenter ou de diminuer l'importance des meilleurs rangs ou d'atténuer la largeur de l'espace entre les individus de plus fort rang et ceux de plus bas rang.

Cette méthode pose comme inconvénient un risque de convergence prématurée à cause de la grande pression exercée par la sélection. Pour éviter ce problème, les auteurs ont introduit une fonction de partage de fitness afin de mieux répartir les solutions le long de la frontière Pareto. Les performances de l'algorithme demeurent toutefois dépendantes de la valeur du paramètre σ_{sh} utilisé dans la fonction de partage.

3.7.2.2 Niched Pareto Genetic Algorithm (NPGA)

Cette méthode proposée par Horn et Napfliotis [1994] utilise une sélection par tournoi basée sur la notion de dominance de Pareto. Elle compare deux individus pris au hasard avec une sous-population de taille t_{dom} également choisie au hasard. Si un de ces deux individus est non dominé par le sous-groupe et l'autre non, il est alors sélectionné. Lorsque les deux individus sont soit non dominés soit dominés par le sous-groupe, une fonction de partage de fitness est appliquée pour déterminer celui qui sera sélectionné. Le paramètre t_{dom} permet d'exercer une pression variable sur la population et ainsi d'augmenter ou de diminuer la convergence de l'algorithme.

Le principal avantage de cette technique est qu'elle ne nécessite pas d'assigner de manière explicite une fitness à un individu. Néanmoins, comme pour le MOGA, la performance de l'algorithme dépend du paramètre σ_{sh} de la fonction de partage et de la valeur de t_{dom}.

3.7.2.3 Non dominated Sorting Genetic Algorithm (NSGA)

Cette méthode a été proposée par Srinivas et Deb [1994]. Comme le MOGA, l'approche NSGA est basée sur le classement non dominé. Elle affecte à chaque individu une fitness factice selon le front auquel il appartient. Chaque front correspond à un groupe d'individus ayant le même degré de dominance au sens Pareto. Les individus du Front 1 auront une meilleure fitness que ceux du Front 2 qui eux, auront une meilleure fitness que ceux du Front 3 et ainsi de suite. Pour la sélection, elle applique ensuite le partage de fitness au niveau de chaque front pour maintenir la diversité. La Figure 3.8 montre le classement d'une population par front dans un contexte de minimisation. L'ensemble des solutions non dominées est représenté par les solutions du front 1.

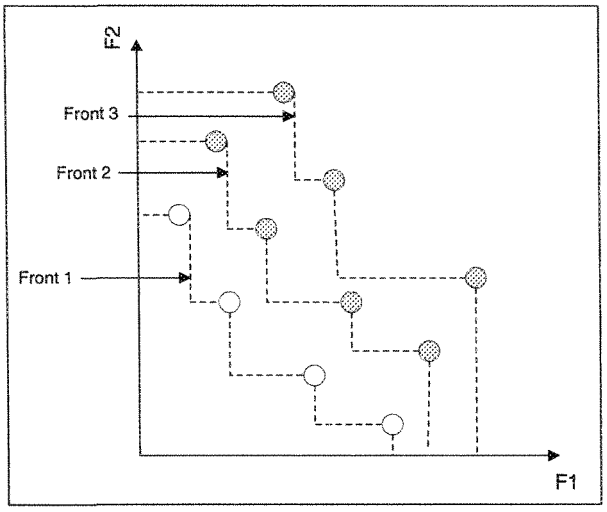

Figure 3.8 : Illustration du classement par front dans le NSGA [Zitzler *et al.* 1999]

Cette méthode paraît moins efficace en temps de calcul que la méthode MOGA car le temps de calcul nécessaire à la construction des fronts et au partage de la fitness est important ($O(nobj|POP|^3)$ avec $nobj$ = le nombre d'objectifs et $|POP|$ = la taille de la

population). L'utilisation d'une fonction de partage sur l'espace des solutions et le tri des solutions en différentes frontières semblent toutefois plus appropriés pour maintenir une grande diversité de la population et pour répartir plus efficacement les solutions sur la frontière Pareto [Berro 2001]. Cependant, comme pour le NPGA et le MOGA, l'efficacité de la méthode dépend de la spécification des paramètres de la fonction de partage.

3.7.3 Les techniques élitistes

À l'opposé des méthodes non élitistes, une technique élitiste comporte une ou plusieurs stratégies permettant de conserver les meilleures solutions trouvées au cours de l'exécution de l'algorithme.

3.7.3.1 NSGAII

Dans cette deuxième version du NSGA [Deb 2000], l'auteur tente de résoudre les critiques faites à la première version de l'algorithme : complexité, utilisation du partage de fitness et non-élitisme.

La complexité de l'algorithme NSGA est notamment due à la procédure de création des différents fronts. Pour diminuer la complexité de calcul du NSGA, Deb propose une modification de la procédure de tri de la population en plusieurs fronts. Au total, cette nouvelle procédure a une complexité de $O(nobj|POP|^2)$.

L'autre critique sur le NSGA concerne l'utilisation de la fonction de partage, méthode qui exige le réglage d'un ou plusieurs paramètre(s) et qui est également forte consommatrice de calculs. Dans le NSGAII, Deb remplace la fonction de partage de fitness par une fonction de remplacement. Pour cela, il attribue deux caractéristiques à chaque individu :

- i_{rank} qui représente le rang de non-dominance de l'individu. Cette caractéristique dépend de la frontière à laquelle appartient l'individu. Les individus du front 1 auront un i_{rank} de 0 car ils sont non dominés. Les individus du front 2 un i_{rank} de 1 car ils ne sont dominés que par des individus du front 1 et ainsi de suite; et
- $i_{distance}$ qui représente la distance de remplacement de l'individu et permet d'estimer la densité de la population autour de lui.

Comme illustré dans la Figure 3.9, pour estimer la densité au voisinage d'une solution i, on calcule la distance moyenne sur chaque objectif, entre les deux points les plus proches situés de part et d'autre de la solution. Cette quantité, appelée $i_{distance}$, sert d'estimateur de taille du plus large hyper-cube incluant le point x sans inclure un autre point de la population. L'algorithme de calcul est de complexité $O(nobj|POP|\log(|POP|))$. Cette distance de remplacement va être utilisée pour guider le processus de sélection.

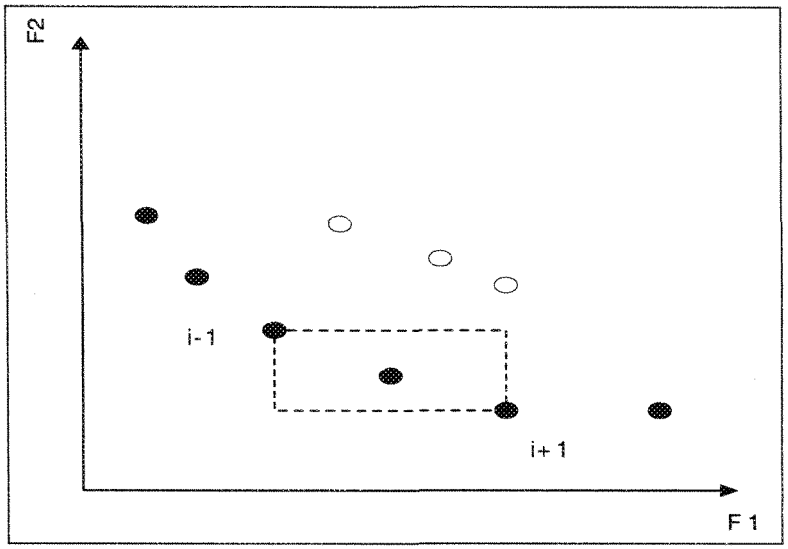

Figure 3.9 : Illustration du calcul de la $i_{distance}$ du NSGAII [Deb 2000]

Pour répondre à la critique de non-élitisme, l'auteur utilise dans cette version une sélection par tournoi et modifie la procédure de passage entre deux générations. Si deux solutions sont sélectionnées pour participer au tournoi, la solution de plus bas rang i_{rang} sera retenue. Si les deux rangs sont identiques, il est préférable de choisir la solution située dans une région dépeuplée, c'est-à-dire avec une valeur $i_{distance}$ importante.

La Figure 3.10 illustre le fonctionnement du NSGAII. À chaque itération t de l'algorithme, une population d'enfant Q_t de taille $|POP|$ est générée à partir d'une population POP_t de parents de taille identique. Les deux populations sont alors combinées dans une population $POP_t \cup Q_t$ de taille $2*|POP|$. La nouvelle population est ensuite triée en plusieurs fronts selon le degré de non-dominance. Lorsque la procédure de tri est complétée, une nouvelle population POP_{t+1} de taille $|Pop|$ est créée à partir des solutions des différents fronts de $POP_t \cup Q_t$. On commence avec les solutions du premier front et ainsi de suite. Comme la taille de $POP_t \cup Q_t$ est de $2*|Pop|$ et que la taille de la nouvelle population est juste de $|POP|$, tous les fronts ne pourront pas être contenus dans POP_{t+1}. Les fronts qui n'entreront pas dans la nouvelle population seront tout simplement supprimés. Cependant, il se peut qu'il y ait dans un front plus de solutions que de places disponibles. Dans ce cas, les individus du front situés dans les régions les plus isolées ($i_{distance}$ importante) sont choisis et les autres sont éliminés.

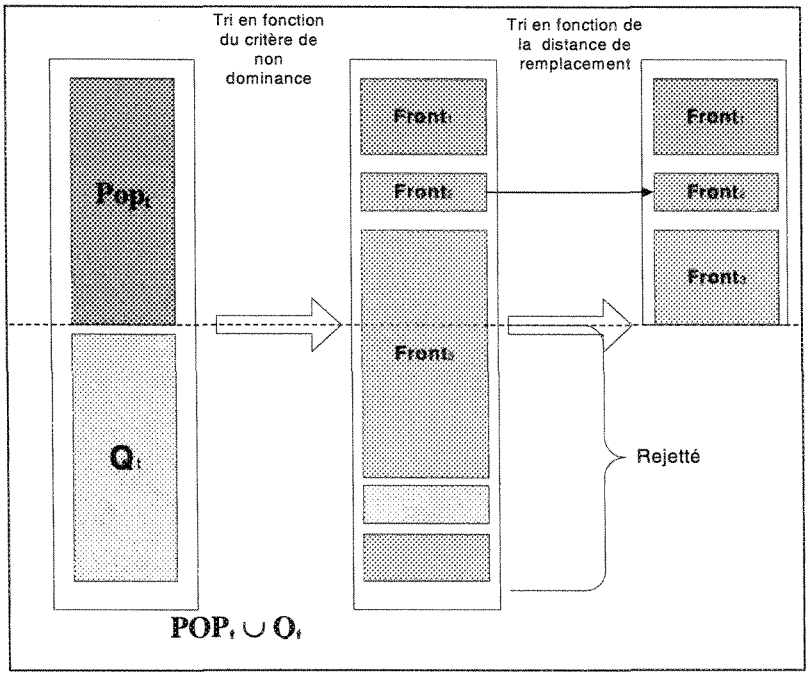

Figure 3.10 : Illustration du fonctionnement du NSGAII [Deb 2001]

Cette nouvelle version de NSGA a permis de réduire la complexité de l'algorithme à $O(nobj|POP|^2)$, de créer une méthode plus élitiste et de supprimer les paramètres de la fonction de partage de fitness. Cependant, il y a un risque de perte de solutions lorsque le nombre de solutions PO dépasse la taille de la population.

3.7.3.2 Strength Pareto Evolutionary Algorithm (SPEA) et SPEA2

Proposé par Zitler *et al.* [1999], le SPEA est basé simultanément sur les concepts de non-dominance et d'élitisme. En plus d'une population initiale de taille $|POP|$, le SPEA utilise une population externe, appelée archive, pour maintenir les solutions PO. Dans cette méthode, le passage d'une génération à une autre commence par la mise à jour de l'archive. Tous les individus non dominés sont copiés dans l'archive et les individus dominés déjà présents sont supprimés. Si le nombre d'individus dans l'archive excède un nombre donné,

on applique une technique de regroupement (*clustering*) pour réduire l'archive. La fitness de chaque individu est ensuite mise à jour avant d'effectuer la sélection en utilisant les deux ensembles. Pour terminer, on applique les opérateurs génétiques de croisement et de mutation.

Une version révisée du SPEA a été récemment proposée pour corriger les lacunes de la version précédente : le SPEA 2 [Zitzler *et al.* 2001]. L'algorithme du SPEA 2 comporte trois principales différences avec son prédécesseur : changement au niveau du calcul de la fitness, introduction d'une nouvelle technique d'estimation de la densité et d'une nouvelle technique pour la réduction de l'archive.

Pour éviter que les individus dominés par les mêmes membres de l'archive aient des fitness identiques, le mécanisme permettant le calcul de la fitness du SPEA 2 tient compte à la fois du nombre d'individus dominés par une solution et du nombre d'individus qui domine la solution. Autrement dit, pour chaque individu x de l'archive $\overline{POP_t}$ de taille $|\overline{POP_t}|$ et de la population POP_t de taille $|POP|$, on calcule la force $S(x)$ représentant le nombre de solutions qu'il domine. La fitness brute $R(x)$ d'un individu x correspondant à la somme des forces des individus qui dominent x.

Bien que la fitness brute soit une technique de nichage basée sur la notion de dominance, elle se révèle inefficace lorsque la plupart des individus sont des solutions non dominées [Zitzler *et al.* 2001]. C'est pourquoi les auteurs ont introduit une technique d'estimation de la densité basée sur le concept du $k^{\text{ième}}$ plus proche voisin [Silverman 1986]. Pour cela, la distance entre un individu x et tous les individus y de l'archive et de la population est calculée et stockée dans une liste. Après avoir trié la liste en ordre croissant, le $k^{\text{ième}}$

élément donne la distance σ_x^k recherchée. Dans l'algorithme, les auteurs utilisent $k=\sqrt{|Pop|+|\overline{Pop}|}$. La densité *Dens(x)* correspondant à *x* est donc :

$$Dens(x) = \frac{1}{\sigma_x^k + 2} \qquad (3.11)$$

Au dénominateur, on ajoute deux pour s'assurer que celui-ci sera supérieur à zéro et que $Dens(x) < 1$. Finalement, en ajoutant *Dens(x)* à la fitness brute *R(x)* on obtient la fitness *f(x)* de l'individu *x*.

Pour réduire les pertes de solutions dues à l'ancienne méthode de regroupement, les auteurs ont introduit une nouvelle technique de réduction de l'archive. La Figure 3.11 représente le principe de réduction de la taille de l'archive du SPEA 2. Sur la gauche, un ensemble de solutions non dominées est présenté. Sur la droite, on montre quelles solutions seront enlevées de l'archive et dans quel ordre par la fonction de réduction. Si la taille de l'archive doit contenir seulement cinq éléments, la fonction de réduction supprimera en premier la solution1 puis la solution 2 et enfin la solution 3.

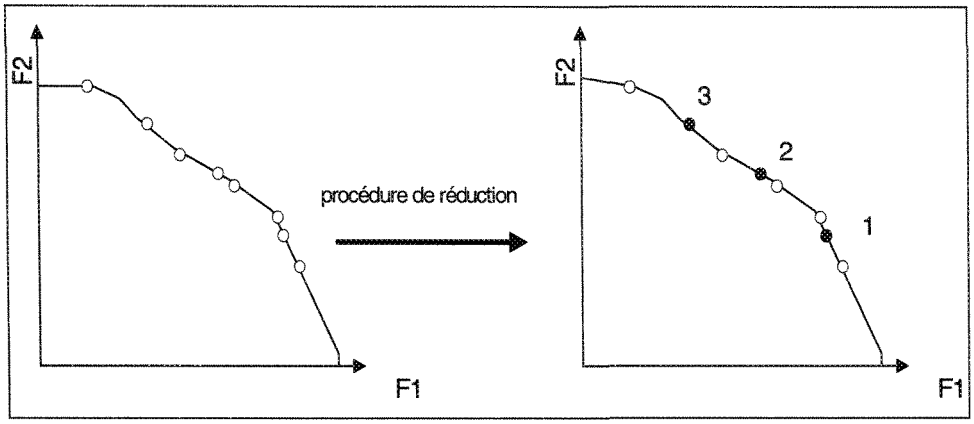

Figure 3.11 : Illustration de la méthode de réduction de l'archive utilisée par le SPEA 2 [Zitzler *et al.* 2001].

Cette nouvelle version du SPEA a permis de combler certaines lacunes de son prédécesseur. Cependant, les performances de l'algorithme dépendent généralement de la taille de l'archive.

3.7.3.3 Pareto Archived Evolution Strategy (PAES) et Memetic- PAES (M-PAES)

Le PAES est une méthode initialement développée comme une méthode de recherche locale pour un problème de routage d'information off-line. Les premiers travaux de Knowles et Corne [1999] ont montré que cette méthode uni-objectif fournissait des résultats supérieurs aux méthodes de recherche basées sur une population. Ces performances ont poussé les auteurs à adapter cette technique à la résolution de problèmes multi-objectifs. Les particularités de cette méthode sont les suivantes :

- elle n'est pas basée sur une population. Elle n'utilise qu'un seul individu à la fois pour la recherche des solutions;
- elle utilise une population annexe de taille déterminée permettant de stocker les solutions temporairement PO;

- l'algorithme utilisé est très simple et inspiré d'une *stratégie d'évolution* (*1+1*) [Rechenberg 1973]; et
- elle utilise une technique de *crowding* basée sur un découpage en hyper-cubes de l'espace des objectifs.

Le M-PAES est une extension de la recherche locale (*1+1*)-PAES combinée avec l'utilisation d'une population *POP* [Knowles et Corne 2000]. Dans cette approche, les optima locaux trouvés par la procédure (*1+1*)-PAES sont périodiquement combinés par croisement. La mise à jour de l'archive s'avère cependant plus complexe que celle utilisée par le PAES. En effet, dans l'algorithme de PAES l'archive permet de mémoriser les solutions non dominées trouvées jusque là et sert aussi comme ensemble de comparaison permettant d'évaluer la dominance des nouveaux individus. Pour effectuer ces deux rôles, le M-PAES a besoin de deux archives : une archive globale pour mémoriser un ensemble de solutions non dominées et une archive locale utilisée comme échantillon de comparaison lors des recherches locales.

Les tests effectués par les auteurs montrent que M-PAES est supérieure dans tous les cas à (*1+1*)-PAES. Mais la comparaison avec SPEA a produit des résultats équivalents bien qu'il soit difficile de comparer ces deux algorithmes.

3.7.3.4 Pareto Envelope Based Selection Algorithm (PESA) et PESAII (Region Based Selection)

Également proposé par Knowles et Corne [2000], le PESA reprend approximativement le principe de crowding développé dans PAES et définit un paramètre appelé *squeeze_factor* qui représente la mesure d'encombrement d'une zone de l'espace. Alors que

PAES est basé sur une stratégie d'évolution, PESA est une méthode basée sur les algorithmes génétiques qui définit deux paramètres concernant la taille des populations d'individus : POP_I (taille de la population interne) et POP_E (taille de la population externe ou archive).

Au cours de la phase de recherche, une solution courante de Pop_I peut entrer dans l'archive POP_E si elle est non dominée dans POP_I et si elle est non dominée dans POP_E. Une fois la solution insérée dans l'archive, on supprime tous les membres de l'archive qu'elle domine. Si l'ajout crée un dépassement de capacité de POP_E alors le membre de l'archive ayant le *squeeze_factor* le plus élevé est supprimé. Le paramètre *squeeze_factor* est égal au nombre d'individus qui appartiennent au même hyper-cube. Il est utilisé comme mesure de fitness des individus qui appartiennent à cette zone. Contrairement au PAES où la mesure d'encombrement n'est utilisée que pour la mise à jour de l'archive, le *squeeze_factor* est aussi utilisé lors du processus de sélection dans l'algorithme de PESA. Lors de la sélection par tournoi, le candidat choisi sera celui qui a le plus petit *squeeze_factor*. Ainsi, les recherches seront orientées vers des zones de la frontière de Pareto qui sont le moins représentées dans la population courante.

Récemment, Corne [2001] propose une nouvelle version de l'algorithme PESA basée sur l'utilisation d'hyper-cubes dans l'espace des objectifs. Au lieu d'effectuer une sélection en fonction de la fitness des individus comme dans PESA, cette méthode effectue une sélection par rapport aux hyper-cubes occupés par au moins un individu. Après avoir sélectionné l'hyper-cube, on choisit aléatoirement l'individu dans l'hyper-cube. Cette méthode se montre plus efficace que son prédécesseur à répartir les solutions sur la

frontière de Pareto. Cela est dû à sa capacité de choisir avec une plus grande probabilité que le tournoi classique, des individus situés dans des zones peu explorées.

3.7.3.5 Micro-genetic Algorithm (Micro-Ga)

Coello trouve que les recherches actuelles n'accordent pas assez d'importance à l'efficience des méthodes d'optimisation multi-objectifs. L'auteur propose alors une méthode basée sur une population avec un nombre très faible d'individus [Coello Coello et Pulido 2001] afin de limiter les temps de calculs. Cette technique se base sur les résultats théoriques obtenus par Goldberg [1989], selon lesquels une taille de population très faible suffit à obtenir une convergence indépendamment de la longueur du chromosome. Le mécanisme suggéré par Goldberg est d'utiliser une population très petite et d'appliquer les opérateurs génétiques jusqu'à obtention d'une convergence (identifiée par une perte de diversité génétique). Ensuite, on crée une nouvelle population en copiant les meilleurs individus de la population précédente, le reste de la population étant générée aléatoirement.

Coello applique ce mécanisme aux problèmes d'optimisation multi-objectifs en utilisant un algorithme génétique avec une petite taille de population associée à une archive et une heuristique de distribution géographique. Il définit une population extérieure divisée en deux parties : une partie remplaçable et une partie non remplaçable. La portion non remplaçable ne change pas durant le processus de modification et sert à maintenir la diversité. Elle ne sera mise à jour que lorsque le Micro-GA aura convergé. La portion remplaçable est totalement modifiée à intervalle régulier. Ce dernier est défini en nombre de cycles du Micro-GA.

Au début de l'algorithme du Micro-GA, la population est constituée à l'aide d'individus sélectionnés aléatoirement dans la population externe. Ensuite, l'algorithme se déroule de manière classique. En fin de cycle, lorsque la population du Micro-GA a perdu sa diversité, deux individus non dominés sont sélectionnés pour mettre à jour l'archive. L'approche utilisée est similaire à celle de PAES. Ensuite, ces deux mêmes individus sont comparés à deux individus de la partie non remplaçable. Si l'un des deux premiers domine l'un des deux autres alors, il le remplace dans la partie non remplaçable.

3.7.3.6 Pareto Memetic Strategy for Multiple Objective optimization (PMS^{MO})

Proposé récemment par Zinflou *et al.* [2006], le PMS^{MO} est un algorithme mimétique élitiste combinant les concepts de dominance Pareto, de niche et d'élitisme avec des opérateurs de recherche locale.

En plus d'une population initiale *POP*, le PMS^{MO} utilise deux archives : une *archive locale,* notée *A* et une *archive globale*, notée Ă. L'archive locale consiste en une population externe de taille déterminée et fixée à $|A|$ permettant de stocker les solutions temporairement PO qui participent au processus de sélection. Lorsque le nombre de solutions non dominées dans l'archive locale dépasse $|A|$, une procédure de réduction de l'archive est appliquée afin de ne conserver que les individus situés dans les régions les plus isolées. Les autres individus sont, quant à eux, transférés dans l'archive globale et ne participent plus au processus de sélection.

Le PMS^{MO} utilise deux paramètres pour calculer la fitness à un individu : le *facteur de dominance* R^+ et le *facteur d'isolement D*. Le *facteur de dominance* est déterminé dans l'algorithme à l'aide de l'équation suivante :

$$R^{+}(x)=\begin{cases} \dfrac{S(x)}{1+2*S(x)} & si \quad \sum_{y\in Pop_t\cup A_t,\ y\succ x} S(y)=0 \\ \sum_{y\in Pop_t\cup A_t,\ y\succ x} S(y) & sinon \end{cases} \qquad (3.12)$$

Pour les solutions non dominées, cette méthode de calcul du *facteur de dominance* permet de mieux tenir compte de la distribution des solutions dominées dans l'espace de recherche en favorisant les solutions non dominées situées dans des régions sous exploitées. Le *facteur d'isolement* quant à lui est calculé selon le concept du $k^{\text{ième}}$ plus proche voisin [Silverman 1986] de façon similaire au SPEA 2. À la seule différence que dans le PMS^{MO}, la distance d'un individu x de l'archive locale est calculée seulement par rapport aux autres individus y de la même archive.

Après avoir été généré par croisement, chaque nouvel individu est amélioré à l'aide d'une procédure de recherche locale. Au cours de cette procédure, un mouvement d'une solution x vers une solution voisine y est effectué dans deux cas. Tout d'abord, si la solution voisine y domine la solution de départ x alors un mouvement est effectué. Si, au contraire, aucune relation de dominance ne peut être identifiée entre x et y alors, à chaque génération, un poids w_{obj} est déterminé aléatoirement pour chacun des objectifs *obj* du problème à résoudre. La somme des poids w_{obj} étant égale à 1. La valeur normalisée *vn* est ensuite calculée selon la formule :

$$vn = \prod_{obj=1}^{nobj} \left(\frac{f_{obj}(y)}{f_{obj}(x)} \right)^{w_{obj}} \qquad (3.13)$$

où *nobj* représente le nombre total d'objectifs. Si $vn<1$, pour un contexte de minimisation, on effectue un mouvement. Dans le cas contraire, aucun mouvement n'est effectué.
Les tests effectués par les auteurs montrent que le PMS^{MO} est une approche compétitive par rapport au NSGAII et au SPEA2 pour le problème de sac à dos multidimensionnel ainsi que pour un problème industriel de planification de coulée de l'aluminium.

3.8 La mesure des performances des algorithmes multi-objectifs

Comment comparer deux ensembles de solutions pour un problème multi-objectifs ? Intuitivement, la première réponse qui vient à l'esprit serait de considérer chaque ensemble un à un. Toutefois, en optimisation multi-objectifs, cette tâche n'est pas aisée. En effet, plusieurs aspects entrent en ligne de compte comme, par exemple, la qualité (la proximité par rapport à la frontière Pareto théorique), la distribution (est-ce que toutes les parties de la frontière Pareto réelles sont découvertes), la répartition (est-ce que les points sont répartis de manière homogène tout au long de la frontière) et autres. En fait, il est très difficile, voire même impossible, de prendre en compte tous ces paramètres au travers d'une seule valeur numérique [Barichard 2003]. C'est pourquoi il est courant d'utiliser plusieurs mesures (ou métriques) pour tester tel ou tel aspect de l'ensemble obtenu.

De plus, il faut distinguer deux types de métriques : les métriques relatives, qui comparent deux ensembles, et les métriques absolues, qui évaluent un ensemble sans avoir besoin d'autres points ou ensemble de référence. Dans la suite de cette section, nous allons présenter quelques mesures couramment utilisées. Nous mettrons en avant leurs points forts

et leurs points faibles, de manière à aiguiller le choix du développeur quant à la métrique qu'il peut utiliser.

3.8.1 La métrique d'espacement (*Sp*)

La première métrique présentée dans cette partie du document est la mesure d'espacement (spacing metric) introduite par Schott [1995]. Cette métrique permet de mesurer l'uniformité de la répartition des points composant la surface de compromis. Cette métrique peut se définir de la manière suivante. Soit *A* un ensemble de *n* éléments de dimensions *nobj,* alors *Sp(A)* est définie par :

$$Sp(A) = \left[\frac{1}{n-1} * \sum_{i=1}^{n} \left(\overline{dist} - dist_i \right)^2 \right]^{1/2} \qquad (3.14)$$

avec $dist_i = \min_j \left(\left| f_1^i(\vec{x}) - f_1^j(\vec{x}) \right| + \ldots + \left| f_{nobj}^i(\vec{x}) - f_{nobj}^j(\vec{x}) \right| \right)$ $i,j \in [1,\ldots, n]$ *et* $i \neq j$, $\overline{dist}$: moyenne de tous les $dist_i$; et n : nombre d'éléments de l'ensemble de solutions. Pour illustrer la métrique d'espacement, nous reprenons ici un des exemples donnés dans Collette et Siarry [2002]. Cet exemple considère une surface de compromis composée de trois points. Le tableau des distances entre les points de la surface de compromis est représenté au Tableau 3.1.

Point *i*	**Point *j***	**Distance**
(2.5, 9)	(3, 6)	3.5
(2.5, 9)	(5,4)	7.5
(3, 6)	(2.5,9)	3.5
(3, 6)	(5,4)	4
(5,4)	(2.5,9)	7.5
(5,4)	(3, 6)	4

Tableau 3.1 : Exemple de calcul de la métrique d'espacement [Collette et Siarry 2002]

On obtient donc les valeurs $dist_i$ suivante :

- pour le point de coordonnées (2.5, 9), $dist_1 = 3.5$;
- pour le point de coordonnées (3, 6), $dist_2 = 3.5$; et
- pour le point de coordonnées (5, 4), $dist_3 = 4$.

On peut ainsi calculer $\overline{dist} = \frac{3.5+3.5+4}{3} = 3.67$. Pour cet exemple, on a donc

$$Sp = \left[\frac{1}{2} * \left((3.67-3.5)^2 + (3.67-3.5)^2 + (3.67-4)^2\right)\right]^{\frac{1}{2}} = 0.288$$

Plus le résultat de la mesure est proche de 0, meilleure est la répartition des points sur la surface de compromis. Cette métrique est un moyen de mettre en avant l'aspect « diversité » d'un algorithme, mais, dans certains cas, l'évaluation peut être biaisée. En effet, si le front Pareto est discontinu, on aura alors un *Sp* élevé à cause des discontinuités introduites par la forme de la surface de compromis. Ainsi, les trous présents dans la surface de compromis influent sur le calcul de la métrique d'espacement et rendent souvent difficile l'interprétation des résultats [Collette et Siarry 2002].

3.8.2 La métrique (C)

La couverture de deux ensembles ou métrique ***C*** a été introduite en 1999 par Ziztler [1999]. C'est une métrique relative qui permet de comparer deux surfaces de compromis *A* et *B*. La valeur ***C*** (*A*, *B*) correspond au pourcentage d'éléments de *B* dominés par au moins un des éléments de *A*. Le calcul de ***C*** (*A*, *B*) s'effectue selon la formule suivante [Zitzler 1999] :

$$C(A,B) = \frac{\left|\left(b \in B \middle| \exists\ a \in A : a \succ b\right)\right|}{|B|} \qquad (3.15)$$

où | | représente la cardinalité de l'ensemble et le symbole $\succ$ indique une relation de dominance de a par rapport à b. Une mesure $\boldsymbol{C}$ (A, B) = 1 signifie que tous les vecteurs de décision de B sont dominés par ceux de A. À l'inverse, $\boldsymbol{C}$ (A, B) = 0, signifie qu'aucun des points de B n'est faiblement dominé par un point de A. Ainsi, plus la valeur $\boldsymbol{C}$ (A, B) se rapproche de 1, meilleure la surface de compromis A est considérée par rapport à B. La métrique C n'est pas symétrique : $\boldsymbol{C}$ (A, B) $\neq$ 1 - $\boldsymbol{C}$ (A, B), il est par conséquent nécessaire de considérer les deux valeurs $\boldsymbol{C}$ (A, B) et $\boldsymbol{C}$ (B, A) pour obtenir une mesure plus fiable des deux surfaces de compromis A et B à comparer.

La métrique $\boldsymbol{C}$ est un outil intéressant de mesure relative. Elle permet de différencier nettement deux surfaces de compromis lorsque d'autres mesures donnent des évaluations trop proches. Cependant, cette métrique doit être utilisée conjointement à d'autres métriques pour ne pas fausser le jugement [Collette et Siarry 2002; Barichard 2003].

3.8.3 L'hyper-volume (H)

Comme la métrique $\boldsymbol{C}$, l'hyper-volume $\boldsymbol{H}$ a été proposé par Zitzler dans sa thèse de doctorat [1999]. Schématiquement, cette métrique calcule une approximation du volume compris sous la courbe formée par les points de l'ensemble à évaluer. Ainsi, lorsque le problème comporte deux critères, ce calcul correspond au calcul d'une aire. Par contre, lorsque le problème comporte trois critères, la valeur calculée est un volume. Formellement, la métrique $\boldsymbol{H}$ se définit de la manière suivante [Zitzler 1999] : soit A un

sous ensemble de n éléments. La fonction $\boldsymbol{H}$ calcule le volume borné par l'union de tous les polytopes $p_1, p_2, \ldots, p_n$, où chaque p_i est formé par l'intersection des hyperplans des x_i par rapport aux axes du repère, pour chaque axe de l'espace des objectifs, il existe un hyperplan perpendiculaire à l'axe et passant par le point $(f_1(x_i), f_2(x_i), \ldots, f_n(x_i))$. Pour le cas à deux dimensions, chaque p_i représente un rectangle défini par les points de coordonnées (0, 0) et $(f_1(x_i), f_2(x_i))$.

Ainsi, la métrique $\boldsymbol{H}$ permet de donner une valeur numérique à une surface de compromis composée a priori de plusieurs points. Il est clair que la valeur calculée ne permet de mesurer qu'un aspect de la surface de compromis et que, dans certains cas, comme les problèmes non convexes [Van Veldhuizen 1999], la valeur calculée peut être erronée. Un des principaux avantages de cette métrique réside dans le fait qu'elle ne nécessite aucun autre point ou ensemble de référence [Zitzler 1999]. Toutefois, son principal inconvénient est qu'elle nécessite souvent des temps de calcul important [Barichard 2003]. En effet, pour calculer l'union des hyper-volumes induits par chaque point de la surface de compromis, il est nécessaire de calculer un grand nombre d'intersections de volumes. Cette opération, très rapide dans le cas d'un problème à deux objectifs, est beaucoup plus longue lorsque le nombre d'objectifs dépasse trois.

3.8.4 La différence de couverture (*Diff*)

La dernière métrique présentée dans ce chapitre est la différence de couverture entre deux ensembles ou métrique ***Diff***. Cette métrique, aussi introduite par Zitzler [1999], permet de surmonter certains inconvénients de la métrique $\boldsymbol{C}$. Formellement, la métrique ***Diff*** se définit comme suit [Zitzler 1999] : soit deux ensembles A et B, la fonction ***Diff***

définie par ***Diff***$(A,B) = \boldsymbol{H}(A+B) - \boldsymbol{H}(B)$ indique la taille de l'espace dominé par A et pas par B dans l'espace des solutions. Comme pour la métrique **C**, pour comparer deux ensembles *A* et *B* entre eux, il est nécessaire de considérer ***Diff***(A,B) et ***Diff***(A,B). Toutefois, la métrique ***Diff*** étant basée sur l'hyper-volume, elle nécessite souvent des temps de calculs plus importants que ceux de la métrique ***C***.

MCours.com