N° d'ordre : 07/2023-D/MT

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIE DE HOUARI BOUMEDIENNE
FACULTÉ DES MATHÉMATIQUES

THÈSE DE DOCTORAT EN SCIENCES
PRÉSENTÉE POUR L'OBTENTION DU GRADE DE DOCTEUR.

EN MATHÉMATIQUES
SPÉCIALITÉ : PROBABILITÉS-STATISTIQUES

Présentée Par : Massika IKHLEF

Thème

MÉTAHEURISTIQUES ÉVOLUTIONNAIRES MULTIOBJECTIF ET LEUR UTILISATION EN FINANCE.

Soutenu publiquement le 29/04/2023, devant le jury composé de :

M. A. TATACHAK	Professeur à l'USTHB/ FMT	Président
M. M. AIDER	Professeur à l'USTHB/ FMT	Directeur de Thèse
Mme. O. SADKI	Professeurà l'USTHB/ FMT	Examinatrice
M. A. MERAKEB	Professeur à l'UMMTO	Examinateur
M. M. AIDENE	Professeur à l'UMMTO	Examinateur
Mme. C. ADICHE	MC/A à l'U.Boumerdes	Examinatrice

Table des matières

Table des figures

Liste des tableaux

Remerciements.

Je remercie Dieu, pour m'avoir donné la volonté et la force pour accomplir ce modeste travail, EL HAMDOU LI ALLAH.

Je tiens à exprimer ma profonde gratitude au professeur M. Aïder, mon directeur de thèse, et mon guide, pour la confiance qu'il m'a faite en acceptant de diriger mes recherches, et pour ses précieux conseils et orientations, ainsi que pour l'intérêt particulier qu'il a accordé à ce travail. Je ne le remercierai jamais assez pour la grande contribution et l'aide qu'il m'a apporté pour l'aboutissement de ce travail... Merci Mr Aïder.

J'adresse mes remerciements au professeur A. TATACHAK, pour l'honneur qu'il m'a fait en acceptant de présider le jury et d'évaluer ce travail.

Je remercie tous les membres de jurry : O.SADKI, C.ADICHE, M.AIDEN, A.MERAKEB pour avoir accepté d'examiner et d'évaluer ce travail.

Je remercie également mes parents notamment ma chère mère et tous les membres de famille.

Je remercie mes amis et mes collègues et tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

Dédicaces.

Je dédie ce travail :

A mes très chers parents notamment ma chère mère.
A mes chèrs enfants
A ma sœur Naïma.
A mes frères : Saïd et Chafik et à mon époux.
A la mémoire de ma grande mère.
A toute la famille Chikh.

Résumé

Résumé

Dans ce travail, on a proposé une approche métaheuristique évolutionnaire multi-objectifs basée sur la métaheuristique Pareto Ant Colony Optimization (P-ACO) et les algorithmes de tri génétique non-dominés (NSGA II et NSGA III) pour la résolution d'un problème d'optimisation de portefeuille bi-objectif. P-ACO est utilisé pour sélectionner les meilleurs actifs composant le portefeuille efficient. Ensuite, NSGA II et NSGA III sont utilisés séparément pour trouver les poids proportionnels du budget alloué au portefeuille sélectionné. Les résultats obtenus par ces deux algorithmes ont été comparés pour désigner l'algorithme le plus performant. Une autre comparaison est réalisée entre les résultats obtenus et ceux d'une méthode exacte utilisée pour le même problème. Les expériences numériques effectuées sur un ensemble d'instances issues de la littérature ont révélé que la combinaison de la métaheuristique d'optimisation par colonies de fourmis et de l'algorithme génétique NSGA III proposée donne le plus souvent de bien meilleurs résultats que la combinaison de la métaheuristique d'optimisation par colonies de fourmis et de l'algorithme génétique NSGA II.

Abstract

In this paper, we propose a multi-objective evolutionary metaheuristic approach based on the Pareto Ant Colony Optimization (P-ACO) metaheuristic and the non-dominated genetic sorting algorithms (NSGA II and NSGA III) to solve a bi-objective portfolio optimization problem. P-ACO is used to select the best assets composing the efficient portfolio. Then, NSGA II and NSGA III are separately used to find the proportional weights of the budget allocated to the selected portfolio. The results we obtained by these two algorithms were compared to designate the best performing algorithm. Finally, we performed another comparison between our results and those of an exact method used for the same problem. The numerical experiments performed on a set of instances from the literature revealed that the combination of the ant colony optimization metaheuristic and the NSGA III genetic algorithm that we proposed most often gave much better results than both the combination of the ant colony optimization metaheuristic and NSGA II on the one hand and the iterative approach on the other hand.

Introduction générale

L'une des directions les plus importantes de la finance est la théorie de la sélection de portefeuille. L'allocation des actifs d'une manière optimale dans un portefeuille est l'objectif de chaque investisseur ou société financière, alors l'optimisation de portefeuille ou le choix de portefeuille optimal d'actifs financiers est un sujet d'intérêt particulier dans la recherche en mathématiques financières.

L'optimisation du portefeuille d'investissement est le processus d'optimisation de la proportion de capital des actifs détenus pour s'adapter à diverses contraintes ; il donne le rendement le plus élevé avec le moins de risque, alors l'optimisation du portefeuille consiste à choisir le meilleur parmi l'ensemble d'opportunités de titres pour équilibrer l'objectif de maximiser le rendement tout en minimiser le risque.

Markowitz à été le premier à introduire la méthodologie d'optimisation de portefeuille de base en 1952. Le modèle de Markowitz ou le modèle de moyenne-variance (MV) a formulé le problème d'optimisation comme un problème de programmation quadratique dans lequel la fonction de risque était mesurée par la variance des rendements de portefeuille observés autour de leur moyenne , on a supposé que le les rendements du portefeuille sont normalement distribués. Les solutions du problème de Markowitz pour différents niveaux de rendement forment la frontière efficiente qui représente le compromis optimal entre le risque et le rendement.

Cependant, le modèle de moyenne-variance (MV) contrairement à sa réputation théorique, plusieurs critiques ont été adressées à ce modèle comme le choix de la variance en tant que mesure de risque, quelques difficultés lorsqu'il s'agit de traiter un grand nombre d'actifs ou de résoudre le problème de programmation quadratique et de calculer la matrice de variance-covariance (la charge de calcul).

D'autres chercheurs ont proposé des algorithmes heuristiques suffisamment efficaces pour résoudre les problèmes de sélection de portefeuille, Ces algorithmes permettent de sélectionner les meilleurs actifs ainsi que les proportions du capital investies qui rendent le portefeuille sélectionné optimal.

K.F. Doerner en 2004 a proposé un algorithme heuristique basant sur les algorithmes de colonies de fourmis pour la résolution du problème d'optimisation de portefeuille sous contraintes.
John Holland s'est intéressé à l'application de la sélection naturelle à la machine apprenante

(Learning Machine). Il a développé une technique qui permettait aux programmes informatiques d'imiter le processus d'évolution, à la fin des années 60. Le terme "algorithme génétique" est devenu populaire, après la publication de son livre (Holland, 1975, 1992). (Goldberg, 1989) a publié un livre qui fournissait une solide base de travail pour ce domaine de recherche. Par la suite, l'utilisation de ces algorithmes est devenue de plus en plus populaires dans les problèmes d'optimisation multiobjectif (Coello Coello, 2005).

L'objet de cette thèse est de résoudre un problème d'optimisation de portefeuille sous trois contraintes : la contrainte de budget qui signifie que la totalité du budget est répartie sur l'ensemble des actifs sélectionnés, en d'autres termes, la somme des parts investies dans les différents actifs est égale à cent pour cent . La contrainte de cardinalité , qui fixe le nombre d'actifs à sélectionner, on a aussi la contrainte de pré affectation utilisée pour modéliser les préférences subjectives de l'investisseur.

Pour résoudre ce problème, on a proposé une métaheuristique évolutionnaire multiobjectif basée sur les algorithmes de colonies fourmis multiobjectifs (Pareto Ant Colony Optimization P-ACO),initialement proposés dans [Dorigo, 1992, Dorigo et al. , 1996], et sur les deux versions améliorées de l'algorithme génétiques Non Dominated Sorting genetic algorithm (NSGA II et NSGA III). L'approche P-ACO nous a permis de choisir les meilleurs actifs pour la constitution du portefeuille de telle sorte à maximiser le rendement et minimiser le risque du portefeuille, à trouver le bon équilibre et la meilleure combinaison d'actifs. L'utilisation des algorithmes génétique, permettra de trouver les proportions du capital investies .

Notre travail est organisé comme suit :

Dans le premier chapitre, nous avons traité la Théorie Moderne de Portefeuille : nous avons donné quelques définitions concernant le rendement, le risque et nous avons présenté quelques approches d'optimisation de portefeuille d'actifs financiers existants dans la littérature.

Dans le deuxième chapitre , nous avons présenté le vocabulaire ainsi que quelques définitions utilisées dans le domaine de l'optimisation multiobjectif et on a présenter aussi les méthodes de résolutions de ces problèmes.

Dans le troisième chapitre, on traite essentiellement une métaheuristique multiobjectif à base de colonies de fourmis (Pareto Ant Colony Optimization P-ACO). Cette métaheuristique se base sur une stratégie phéromonale qui permet de converger vers les solutions. Le choix d'une telle stratégie est à la fois déterminant et délicat en fonction des problèmes.

Le quatrième chapitre présente une étude de la littérature dans le domaine des algorithmes évolutionnaires. En présentant, le détail du fonctionnement des deux algorithmes génétiques NSGAII et NSGA III, plus un ensemble de techniques de sélection, croisement et mutation - potentiellement – utilisables, dans un problème d'optimisation évolutionnaire multi objectifs.

Le cinquième chapitre intitulé "métaheuristique évolutionnaire multiobjectif" présente la contribution majeur de cette thèse. Dans ce chapitre on décrit la méthode proposée et on présente les résultats numériques obtenus. On a présenté aussi les résultats des autres chercheurs trouvés pour la résolution d'un problème similaire et ce pour voir le degré d'efficacité de notre approche, sans oublier l'outil statistique qui a été bien présent dans ce chapitre.

Notons que le contenu du chapitre 5 a fait l'objet de la publication suivante dans une revue de catégorie B : Ikhlef.M, Aïder.M, Multiobjective evolutionary metaheuristic approach to the constrained portfolio optimization problem, Pesquisa Operacional (2023) 43 : e266962 P.1-23,doi : 10.1590/0101-7438.2023.043.00266962.

Enfin, nous présentons une conclusion ainsi que des perspectives.

1

Problèmes d'Optimisation de Portefeuille

Sommaire

1.1 Modèle Moyenne-Variance de Markowitz (1952)

La théorie moderne du portefeuille et les modèles de choix de portefeuille est née en 1952 avec la publication de l'article fondateur de Harry Markowitz. et dans le cadre de la théorie de l'espérance d'utilité, aussi d'après l'axiomatisation des préférences individuelles de von Neumann et Morgenstern. et les travaux de Harry Markowitz pour la sélection de titres pour créer le portefeuille le plus efficient possible Ce mode de sélection permet de minimiser le risque pour un niveau de rendement choisi. c'est à dire qui possède la rentabilité maximum pour un niveau de risque minimum.

Le modèle proposé par Markowitz est la recherche d'un portefeuille efficace, il a l'espérance de rentabilité la plus forte parmi les portefeuilles qui ont la même variance de rentabilité que lui. L'ensemble de tous les portefeuilles efficaces constitue la frontière efficiente de Markowitz (appelée la frontière efficace).

L'investisseur est alors présumé prendre ses décisions en fonction seulement de deux paramètres : l'espérance de sa richesse qu'il souhaite la plus grande possible, et sa variance qu'il désire la plus faible possible. Cette approche a offert le premier traitement systématique d'un dilemme auquel les investisseurs font face.

L'approche de Markowitz est plus connue sous le nom de l'approche Moyenne-Variance (MV). Cette approche connaît aussi certains inconvénients. D'abord, elle suppose que, — Soit les rendements suivent une distribution normale (ou multi-normale),
— Soit la fonction d'utilité est quadratique.
— Manque de sensibilité du modèle quant aux différences entre les gains et les pertes.
— L'approche MV n'est pas, en général, compatible avec la dominance stochastique.
— MV n'est pas cohérente car elle ne respecte pas les axiomes de monotonie et d'invariance par translation.

1.1.1 Principales hypothèses du modèle M-V

Commençant par les hypothèses relatives aux actifs financiers.

Hypothèse 01 : Tout investissement est une décision prise dans une situation de risque : le rendement d'un actif financier est une variable aléatoire, distribuée selon une loi normale, c'est- à-dire une distribution symétrique stable entièrement définie par deux paramètres, l'espérance mathématique de return et l'écart-type de la distribution de probabilité de return.

Hypothèse 02 : Les rendements des différents actifs financiers sont corrélés $(Cov(R_i, R_j) = 0)$ ne fluctuent pas indépendamment les uns des autres.

Hypothèse 03 : Les marchés sont parfaits : toutes les conditions pour que les prix correspondent à la réalité du moment sont réunies. L'entrée et la sortie sont libres et sans coût.

Les hypothèses relatives aux comportements des investisseurs sont :

Hypothèse 01 : Le comportement de tous les investisseurs est caractérisé par un degré plus ou moins prononcé d'aversion vis-à-vis du risque. Ce dernier est mesuré par l'écart-type de la distribution de probabilité du rendement.

Hypothèse 02 : Les investisseurs sont rationnels : bien que leur fonction de préférence soit purement subjective, leur comportement est caractérisé par un degré plus ou moins prononcé d'aversion au risque.

Hypothèse 03 : Tous les investisseurs ont la même période de l'investissement, ont prennent leurs décisions en même temps. Cette simplification, qui peut paraître exagérée, permet de mettre en œuvre un modèle de décision qui tient compte du caractère hautement combinatoire du portefeuille.

1.1.2 Description du Modèle Mean–Variance de Markowitz

Supposons qu'il y ait n actifs financiers, dont les rendements sont donnés par les variables aléatoires R_1, ..., R_n. Alors le rendement du portefeuille R_P est une variable aléatoire dont l'espérance sera donnée par :

$$E(R_P) = E\left[\sum_{i=1}^{n} R_i x_i\right] = \sum_{i=1}^{n} E(R_i)x_i = \sum_{i=1}^{n} \mu_i x_i \qquad (1.1)$$

avec :

$E(R_i) = \mu_i$: Le rendement espéré pour l'actif numéro i.
x_i sont les proportions budgétaires allouées au portefeuille, telle que $x_i \geq 0 \ \forall i = 1,...,n$ et $\sum_{i=1}^{n} x_i = 1$

Et le risque du portefeuille est donné par :

$$Var(R_p) = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} Cov(R_i, R_j)x_i x_j = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \rho_{ij} x_i x_j \qquad (1.2)$$

Où :

ρ_{ij} est la covariance entre le i^{me} et le j^{me} actif.

Le modèle d'optimisation de Makowitz qui minimise le risque s'écrit comme suit :

$$\min\left[\sum_{i=1}^{n}\sum_{i=1}^{n} x_i x_j \rho_{ij}\right]$$

Sous les contraintes :

$$\begin{cases} \sum_{i=1}^{n} E(R_i) & \geq & \beta \\ \sum_{i=1}^{n} x_i & \doteq & 1 \\ x_i & \geq 0 & \forall i = 1,...,n \end{cases}$$

Avec : β est le rendement minimum attendu par l'investisseur.

1.1.3 Avantages et inconvénients du modèle

L'utilisation de la variance comme mesure de risque a des avantages et des inconvénients. L'avantage de cette mesure de risque est sa simplicité. De plus, en utilisant la matrice de variance-covariance, nous pouvons construire un modèle de programmation quadratique convexe, afin de calculer les portefeuilles efficients. Actuellement, un problème quadratique convexe se résout facilement même pour des dimensions très grandes . Mais cela ne suffit pas car certaines contraintes du monde réel ne sont pas prises en compte dans le modèle de base.

L'approche de Markowitz est plus connue sous le nom de l'approche Moyenne-Variance (MV). Cette approche connaît aussi certains inconvénients. D'abord, elle suppose que :
- les rendements suivent une distribution normale (ou multi-normale),
- la fonction d'utilité est quadratique.

Ces deux cas sont malheureusement assez peu réalistes. La deuxième critique est due au manque de sensibilité du modèle quant aux différences entre les gains et les pertes car l'approche MV pénalise de la même manière les gains ou les pertes s'éloignant trop de la moyenne . L'approche MV n'est pas, en général, compatible avec la dominance stochastique. Finalement, l'approche MV n'est pas cohérente car elle ne respecte pas les axiomes de monotonie et d'invariance par translation.

Après le travail de Markowitz, les modèles alternatifs ont été développé pour pallier aux défauts du modèle MV.

1.2 Le modèle d'équilibre des actifs financiers (MEDAF)

1.2.1 Définition du MEDAF

En 1963. A la suite des travaux de Markowitz, le modèle linéaire vient de pénétrer par effraction dans la finance moderne, en liant l'indice de marché à celui de portefeuille.

Grâce aux travaux de Jack Treynor (1961), William Sharpe John Lintner (1965) et Jan Mossin (1966) basés sur la théorie de choix de portefeuille de Markowitz, Sharpe un an plus tard, qui va rapidement s'apercevoir que l'on peut dire beaucoup mieux et propose le modèle d'équilibre des actifs financiers, ou MEDAF. Ce modèle d'évaluation des actifs financiers (MEDAF) appelé aussi « Capital Asset Pricing Model (CAPM), en anglais » a été développé par Sharpe (1964), Lintner (1965), et Mossin (1966).

(MEDAF) est un modèle dont le but est de déterminer les rentabilités appropriées des différents actifs qui forment le marché financier. C'est un modèle d'équilibre, c'est a dire que les

rentabilités sont expliquées par l'adéquation entre l'offre et la demande en titres des différents agents qui interviennent sur le marché. Le message essentiel du MEDAF est que le risque de chaque titre peut être décomposé en une composante systématique commune à tous les actifs du marché et une composante spécifique diversifiable, le marché ne rémunérant que la composante systématique.

La prise en compte de la pluralité des investisseurs conduit sous certaines hypothèses à un modèle exprimant les rentabilités espérées d'équilibre, le Modèle d'Equilibre des Actifs Financiers (MEDAF), ou Capital Asset Pricing Model (CAPM). Son message central est que, pour tout actif financier pris isolément, la relation entre son risque et sa rentabilité espérée est linéaire, à condition de mesurer ce risque par sa covariance avec le marché pris dans son ensemble, et non sa variance ou son écart-type.

La façon la plus simple (il y en a d'autres) de déterminer la relation entre rentabilité espérée et risque d'un titre financier à l'équilibre du marché est de supposer que tous les investisseurs obéissent au critère M V et qu'il existe un actif sans risque. De plus,on suppose que les individus ont le même horizon d'investissement et des anticipations identiques, c'est-à-dire qu'ils utilisent tous le même vecteur des rentabilités espérées et la même matrice de variance covariance.

1.2.2 Principales hypothèses du MEDAF

Le modèle d'évaluation des actifs financiers (MEDAF), est soumis aux hypothèses suivantes :

Hypothèses Relatives aux Comportements des Investisseurs

- les investisseurs exigent une rentabilité d'autant plus forte que le risque est élevé : il existe donc une relation croissante entre rendement et risque.
- les investisseurs raisonnent tous dans le cadre de Markowitz, avec le même objectif d'optimisation statique,
- les investisseurs disposent de la même information et des mêmes estimations pour les rendements.
- les investisseurs ont la même période de l'investissement.
- les investisseurs prennent leurs décisions en même temps.
- les investisseurs détiennent leurs actifs pendant la même période.

Hypothèses Relatives aux Actifs Financiers

- le marché est efficient.
- il n'y a pas de coûts de transaction.
- Un actif sans risque est disponible.
- Les dividendes et les gains de capitaux ne sont pas taxés.
- Pas d'influence sur les prix par les acheteurs et les vendeurs qui interviennent sur le marché.

Etant donné un portefeuille constitué de n actions de rendements $r_1, r_2, ..., r_n$; et un actif sans risque de rendement r_0.

Le rendement espéré de ce portefeuille est donné par :$\sum_{i=1}^{n} x_i r_i$

Sous ces conditions, un modèle linéaire exprime l'excès de rendement des actifs risqués (par rapport au rendement sans risque) en fonction du rendement RM d'un portefeuille de marché :

$$\forall i \in 1, ..., n, r_i - r_0 = \beta_i(R_M - r_0) + \varepsilon_i \qquad (1.3)$$

avec ε_i un bruit indépendant de R_M. et x_i représente la proportion investie dans l'action A_i pour $i = 1, ..., n$ La relation qui caractérise le modèle d'équilibre des actifs financiers MEDAF est donnée par :

$$\bar{r_i} = r_0 + \beta_i(\bar{R_M} - r_0) \qquad (1.4)$$

Où :
$\bar{r_i}$: le rendement espéré de l'action A_i.
$\bar{R_M}$: le rendement de portefeuille de marché.
σ_M^2 : le risque de portefeuille de marché.
$\beta_i = \frac{\sigma_{iM}}{\sigma_M^2}$.
$\sigma_{iM} = \sum_{j=1}^{n} x_i \sigma_{ij}$

1.3 Le modèle Basé sur La valeur à risque

Le risque est lié à la volatilité du portefeuille d'actifs. Pendant très longtemps, la mesure naturelle du risque a donc été la volatilité. Par exemple, dans le modèle de sélection de portefeuille de Markowitz, l'agent maximise son espérance de gain pour un niveau de risque donné, qui est mesuré par l'écart-type.

Cette vision du risque n'est cohérente que dans un monde gaussien. Cependant, nous savons depuis fort longtemps que l'hypothèse de normalité des rendements des actifs financiers n'est pas vérifiée. Actuellement, la mesure de risque qui est la plus répandue est la valeur en risque **(Value-at-Risk ou** *VaR***)**.

1.3.1 Présentation de *VaR*

Pour expliquer ce qu'est la *VaR*, commençons par prendre un exemple concret. Considérons que nous avons investis 10000*euros* dans notre portefeuille d'actions. Comment avoir une idée de la perte maximale que le portefeuille peut subir d'ici un mois ?
La réponse la plus logique est que nous pouvons perdre tout notre investissement. Néanmoins, un événement de perte totale est vraiment très peu probable. Une réponse plus réaliste serait par exemple qu'en l'absence d'événement exceptionnel, il y a 5% de chance de perdre 1000*euros*. C'est le type de réponse fournit par la *VaR*. De nombreuses définitions de la *VaR* existent, nous en reprenons deux :

Définition 1 (VaR) la VaR d'un portefeuille d'actifs financiers correspond au montant de pertes maximum sur un horizon de temps donné, si l'on exclut un ensemble d'événements défavorables ayant une faible probabilité de se produire.

Particularités

Deux éléments sont communs à toutes ces définitions et il convient de les choisir judicieusement : l'horizon et le niveau de confiance.

L'horizon est influencé par trois facteurs majeurs :
- il doit être adapté à la durée de détention de l'actif ou du portefeuille objet de l'estimation.
- il doit être suffisamment court pour respecter l'hypothèse d'invariance de la composition du portefeuille.
- il doit être suffisamment court pour que la quantité de données disponible (parfois faible) puisse permettre d'estimer une VaR sur cet horizon.

Mais il peut aussi être fixé par des normes réglementaires. Les organismes financiers utilisent une *VaR* à 10 jours mais cet horizon peut atteindre plusieurs mois lorsqu'il s'agit d'obligations.

Le niveau de confiance est quant à lui influencé par deux facteurs :
- il ne doit pas être trop élevé, sinon le risque de réalisation devient suffisamment faible pour être inintéressant en tant qu'indicateur.
- il doit refléter le degré d'aversion des gestionnaires face au risque de réalisation d'événements extrêmes.

Mais tout comme pour l'horizon, le niveau de confiance peut être déterminé par des normes réglementaires. Dans la pratique, la *VaR* est estimée sur la base de niveau de confiance allant de 90 à 99.9%.

Les normes réglementaires imposées par Bâle II exigent que la *VaR* soit calculée par les banques en utilisant un niveau de confiance de 99% et un horizon de 2 semaines soit 10 jours ouvrables.

1.3.2 Approche classique de calcul

Après avoir choisi un horizon et un niveau de confiance, il reste une dernière étape à réaliser avant de procéder à l'estimation de la *VaR*. Il faut en effet récupérer les données sur lesquels sera calculée la *VaR* et les retraiter. On peut calculer la VaR à partir des rendements du portefeuille mais aussi à partir de sa distribution de Profit & Loss (P&L) qui correspond aux pertes ou profits journaliers.

Considérons un portefeuille composé d'une ou plusieurs actions dont on connait les cours journaliers. On note W_t la valeur de ce portefeuille à la date t. La *VaR* de ce portefeuille pour un horizon d'un jour avec une probabilité α correspond à la perte $\Delta W_{(t+1)} = W_{(t+1)} - W_t$ observée

pour le portefeuille avec une probabilité $1-\alpha$ Autrement dit, $VaR_t(\alpha)$ vérifie l'équation :

$$p\left[\Delta w_{t+1} + VaR_t(\alpha) < 0\right] = 1-\alpha$$

$$p\left[\Delta w_{t+1} < -VaR_t(\alpha)\right] = 1-\alpha$$

$$p\left[\Delta w_{t+1} < VaR_t(\alpha)\right] = \alpha$$

Enintroduisant

F_t la fonction de répartition de la variable aléatoire ΔW pour les valeurs connues à la date t et sa fonction inverse F_t^{-1}.
On exprime alors la *VaR* de la façon suivante :

$$VaR_t(\alpha) = F_t^{-1}(\alpha) \qquad (1.5)$$

En général, on essayera de se ramener à une loi de distribution standard qui pourra être assimilée à la loi des rendements. Pour cela on réduit et on centre ΔW_{t+1} (théorème central limite). Ceci nous donne :

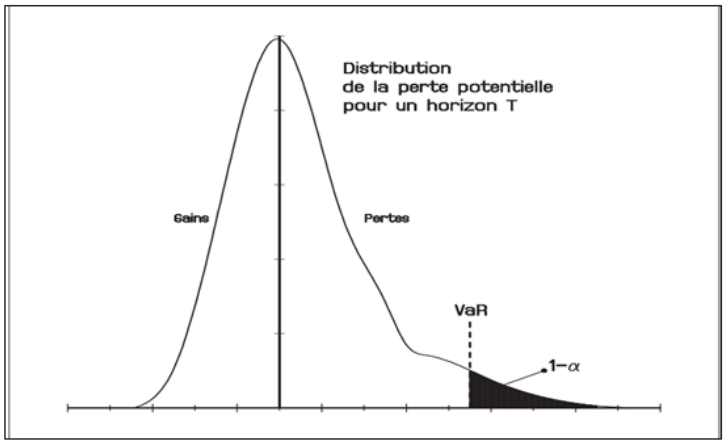
$$\left[\frac{\Delta W_{t+1} - E(\Delta W)}{\sigma(\Delta W)} \leq \frac{VaR_t(\alpha) - E(\Delta W)}{\sigma(\Delta W)}\right] = \alpha$$

On posant : $z_\alpha = \frac{Var_t(\alpha) - E(\Delta W)}{\sigma \Delta W} = \alpha$ où z_α où représente le quantile d'ordre α de la distribution Normale standard, on exprime alors la VaR de la façon suivante :

$$VaR(\alpha) = z_\alpha\, \sigma(\Delta W) + E(\Delta W) \qquad (1.6)$$

La plus part du temps, la loi de distribution des rendements est proche d'une loi Normale. Les tables statistiques donnent des valeurs de z_α pour différentes valeurs de α :

Voici un exemple représentant le positionnement de z_α pour une loi Normale avec $\alpha = 95\%$:



En réalisant ces calculs sur une distribution journalière des rendements, on exprime une *VaR* à horizon d'un jour. Pour passer d'un horizon d'un jour à un horizon de X jours, on utilisera la formule suivante :

$$VaR_X = VaR_1\sqrt{X} \qquad (1.7)$$

Considérons :
— Un seuil de risque de α, équivalent à un seuil de confiance $(1-\alpha)$
— Un horizon temporel N.

Il s'agit donc de la perte maximale potentielle qui ne devrait être atteinte avec une probabilité donnée (ou un risque donné) sur un horizon temporel donné.

Mathématiquement, la notion de la Value-at-Risk se traduit ainsi :

$$Pr(\Delta V < VaR) = 1-\alpha$$

Avec :

ΔV : la variation de la valeur V du portefeuille sur la période de détention.
α : le niveau de confiance.

Pour le modèle mathématique à optimiser, on exprime le risque en fonction de la VaR.

2

Optimisation Multiobjectif

Sommaire

2.1 Qu'est-ce qu'un problème d'optimisation ?

Un problème d'optimisation se définit comme la recherche du minimum ou du maximum (de l'optimum donc) d'une fonction donnée. On peut aussi trouver des problèmes d'optimisation pour lesquels les variables de la fonction à optimiser sont contraintes d'évoluer dans une certaine partie de l'espace de recherche. Dans ce cas, on a une forme particulière de ce que l'on appelle un problème d'optimisation sous contraintes. Ce besoin d'optimisation vient de la nécessité de l'ingénieur de fournir à l'utilisateur un système qui réponde au mieux au cahier des charges. Ce système devra être calibré de manière à :[18]

- occuper le volume minimum nécessaire à son bon fonctionnement (coût des matières premières),
- consommer le minimum d'énergie (coût de fonctionnement),
- répondre à la demande de l'utilisateur (cahier des charges). Mathématiquement parlant, un problème d'optimisation se présentera sous la forme suivante [18] :

$$\left|\begin{array}{ccc} \text{"}minimiser\text{"} & f(\vec{x}) & \\ avec & \vec{g}(\vec{x}) & \leq 0 \\ et & \vec{h}(\vec{x}) & = 0 \end{array}\right.$$

On $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ et $\vec{h}(\vec{x}) \in \mathbb{R}^p$

Ici, les vecteurs $\vec{g}(\vec{x})$ et $\vec{h}(\vec{x})$ représentent respectivement m contraintes d'inégalité et p contraintes d'égalité.

2.2 Essentiels et définitions

Définition 2 (fonction objectif) C'est le nom donné à la fonction f (on l'appelle encore fonction de coût ou critère d'optimisation). C'est cette fonction que l'algorithme d'optimisation va devoir optimiser (trouver un optimum).

Définition 3 (Variables de décision) Elles sont regroupées dans le vecteur $\vec{x}$. C'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction f .

Définition 4 (Minimum global) Un point $\vec{x}^*$ est un minimum global de la fonction f si on a : $f(\vec{x}^*) < f(\vec{x})$ quel que soit $\vec{x}^* \neq \vec{x}$.

Définition 5 (Minimum local fort) Un point $\vec{x}^*$ est un minimum local fort de la fonction f si on a : $f(\vec{x}^*) < f(\vec{x})$ quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$ où $V(\vec{x}^*)$ définit un voisinage de $\vec{x}^*$.

Définition 6 (minimum local faible) Un point $\vec{x}^*$ est un minimum local faible de la fonction f si on a : $f(\vec{x}^*) \leq f(\vec{x})$ quel que soit $\vec{x} \in V(\vec{x}^*)$ et $\vec{x}^* \neq \vec{x}$ où $V(\vec{x}^*)$ définit un voisinage de $\vec{x}^*$.

2.3 Optimisation multiobjectif

La formulation précédente était relative à un problème dans lequel on recherchait un optimum pour une fonction objectif (f dans l'expression précédente). Cependant, lorsque l'on modélise un problème, on cherche souvent à satisfaire plusieurs objectifs. Par exemple, on veut un système performant et on veut aussi que ce système consomme peu. Dans ce cas, on parle de problème d'optimisation multiobjectif (ou problème d'optimisation multicritère). Celui-ci s'écrit de la manière suivante :

$$\left| \begin{array}{lll} \text{"}minimiser\text{"} & \vec{f}(\vec{x}) & \\ avec & \vec{g}(\vec{x}) & \leq 0 \\ et & \vec{h}(\vec{x}) & = 0 \end{array} \right.$$

Où : $\vec{x} \in \mathbb{R}^n$, $vec f(\vec{x}) \in \mathbb{R}^k$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ et $\vec{h}(\vec{x}) \in \mathbb{R}^p$
Ici, $\vec{f}(\vec{x})$ est le vecteur qui regroupe k fonctions objectif, les vecteurs $\vec{g}(\vec{x})$ et $\vec{h}(\vec{x})$ représentent respectivement m contraintes d'inégalité et p contraintes d'égalité.

Le but que l'on se fixe dans la résolution d'un problème d'optimisation multiobjectif est d'optimiser au mieux les différents objectifs.

Dans un problème d'optimisation multicritère, on rencontre souvent des objectifs contradictoires. Deux objectifs sont contradictoires lorsque la diminution d'un objectif entraîne une augmentation de l'autre objectif.

2.3.1 Solutions non dominées et solutions Pareto Optimales

La plupart des algorithmes d'optimisation multiobjectif utilisent le concept de dominance dans leur recherche. Ici, nous définissons le concept de dominance et les termes apparentés et plusieurs techniques pour identifier des solutions non dominées dans une population finie de solutions.

Définition 7 (Vecteur Objectif Idéal) Le vecteur idéal $\vec{f}^*$ du problème est le vecteur de l'espace des critères dont chaque composante est la solution optimale du problème d'optimisation de la fonction objectif sous les contraintes du problème.
Dans le problème de maximisation, le vecteur idéal $\vec{f}^*$ est le vecteur qui optimise chacune des fonctions objectifs c'est-à-dire : $f_i^* = max(f_i(x)), i = 1, \ldots, k$

Définition 8 (Dominance) Le vecteur $\vec{x}_1$ est dit dominé par un autre vecteur $\vec{x}_2$ si :

- $\vec{x}_1$ est au moins aussi bon que $\vec{x}_2$ dans tous les objectifs.
- $\vec{x}_1$ est strictement meilleur que $\vec{x}_2$ dans au moins un objectif.

Les solutions qui dominent les autres mais ne se dominent pas entre elles sont appelées solutions optimales au sens de Pareto (ou solutions non dominées).

2.3.2 Propriétés de la relation de dominance

La relation binaire de dominance $\prec$ telle qu'elle est définie au dessus ;

a. N'est pas réflexive car une solution ne se domine pas par elle-même.

b. N'est pas symétrique car on a jamais : $\vec{x}_1 \prec \vec{x}_2$ et $\vec{x}_2 \prec \vec{x}_1$.

c. Elle est transitive, car si $\vec{x}_1 \prec \vec{x}_2$ et $\vec{x}_2 \prec \vec{x}_3$ implique que $\vec{x}_2 \prec \vec{x}_3$.

Notons que pour toute paire de solution $\vec{x}_1$ et $\vec{x}_2$ ont une et seulement une des affirmations suivantes est vraie :

a. $\vec{x}_1$ domine $\vec{x}_2$

b. $\vec{x}_1$ est dominé $\vec{x}_2$

c. $\vec{x}_1$ et $\vec{x}_2$ sont équivalentes au sens de dominance.

Par la suite, les solutions équivalentes au sens de la dominance seront parfois évoquées comme solutions équivalentes au sens de Pareto ou, encore, comme solutions Pareto équivalentes.

2.4 Méthodes de résolution

En présence d'un problème d'optimisation concret, le chercheur est confronté à la principale difficulté du choix d'une méthode efficace (lorsqu'elles existent), capable de produire une solution optimale ; ou d'une méthode dont la qualité de la solution est acceptable, au prix d'un temps de calcul raisonnable. Face à ce souci un grand nombre de méthodes ont été développées pour tenter d'apporter une réponse satisfaisante à ces problèmes. Parmi celles-ci, nous distinguons les méthodes dédiées à un problème et les méthodes plus génériques pouvant s'appliquer à un ensemble de problèmes. Dans cette section, nous essayons de faire un état d'art sur les différentes méthodes d'optimisation, on distingue deux grandes classes à savoir méthodes exactes et méthodes approchées.

Les méthodes exactes examinent, souvent de manière implicite, la totalité de l'espace de recherche. Ainsi, elles ont l'avantage de produire une solution optimale lorsqu'aucune contrainte de temps n'est donnée. Néanmoins, le temps de calcul nécessaire pour atteindre une solution optimale peut devenir vite prohibitif. Les méthodes approchées constituent une alternative indispensable et complémentaire. Le but d'une telle méthode n'est plus de fournir une solution optimale au problème donné. Elle cherche avant tout à produire une solution sous-optimale de meilleure qualité possible avec un temps de calcul raisonnable. En général, une méthode approchée examine seulement une partie de l'espace de recherche.

2.4.1 Méthodes Exactes

Les méthodes exactes reposent sur l'utilisation d'algorithmes qui mènent de façon sûre vers la solution optimale. Le principe essentiel de ces méthodes est d'énumérer de manière implicite l'ensemble des solutions de l'espace de recherche. Malgré le temps de calcul important que nécessitent, généralement, ces approches, plusieurs méthodes ont été développées. Les méthodes les plus utilisées dans les travaux de recherche sont :

Méthode de pondération des fonctions objectif

Cette approche du résolution des problèmes d'optimisation multiobjectif est la plus évidente. D'ailleurs, on appelle aussi cette méthode l'approche naïve de l'optimisation multiobjectif. Le but, ici, est de revenir à un problème d'optimisation mono objectif, pour lequel existent de nombreuses méthodes de résolution. La manière la plus simple de procéder, consiste à prendre chacune des fonctions objectif et leur appliquer un coefficient de pondération, ensuite faire la somme pondérée des fonctions objectif. On obtient alors une nouvelle fonction objectif. [18]

$$\left|\begin{array}{llll} "Opt" & f(\vec{x}) & \bigoplus_{i=1}^{k} w_i.f_i(\vec{x}) & \\ & \vec{g}(\vec{x}) & \leq & 0 \\ & \vec{h}(\vec{x}) & = & 0 \end{array}\right.$$

On a $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ et $\vec{h}(\vec{x}) \in \mathbb{R}^p$
Les coefficients de pondération w_i respectent la relation suivante : w_i pour tous les $i \in 1,,k$ et $\bigoplus_{i=1}^{k} w_i = 1$

Méthode de Keeney-Raiffa

Cette méthode utilise le produit des fonctions objectif pour se ramener à un problème d'optimisation mono objectif. L'approche utilisée ici est semblable à celle utilisée dans la méthode de pondération des fonctions objectif. La fonction objectif ainsi obtenue s'appelle la fonction d'utilité de Keeney-Raiffa [keeney r. l., and raiffa h, 1993].

$$\left|\begin{array}{llll} "Opt" & f(\vec{x}) & \bigotimes_{i=1}^{k} w_i.f_i(\vec{x}) & \\ & \vec{g}(\vec{x}) & \leq & 0 \\ & \vec{h}(\vec{x}) & = & 0 \end{array}\right.$$

On a $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ et $\vec{h}(\vec{x}) \in \mathbb{R}^p$
Les coefficients de pondération w_i respectent la relation suivante : w_i pour tous les $i \in 1,,k$ et $\bigoplus_{i=1}^{k} w_i = 1$

Méthode de compromis (L'approche par ε -contrainte)

Une autre façon de transformer un problème d'optimisation multiobjectif en un problème mono objectif, est de convertir $k-1$ des k objectifs du problème en contraintes et d'optimiser séparément l'objectif restant. La démarche est la suivante :

a. On choisit un objectif initial (prioritaire) à optimiser.

b. On choisit un vecteur de contraintes initial.

c. On transforme le problème, en conservant l'objectif prioritaire et transformer les autres objectifs en contraintes d'inégalité.

On appelle aussi cette méthode la méthode de la ε -contrainte. Le problème peut être reformulé de la manière suivante :

$$\left|\begin{array}{lrcl} \text{"}Opt\text{"} & f_i(\vec{x}) & & \\ & & & \\ & f_1(\vec{x}) & \leq & \varepsilon_1 \\ & f_2(\vec{x}) & \leq & \varepsilon_1 \\ & \vdots & & \\ & f_{i-1}(\vec{x}) & \leq & \varepsilon_1 \\ & f_{i+1}(\vec{x}) & \leq & \varepsilon_1 \\ & \vdots & & \\ & f_k(\vec{x}) & \leq & \varepsilon_1 \\ & \vec{g}(\vec{x}) & \leq & 0 \\ & \vec{h}(\vec{x}) & = & 0 \end{array}\right.$$

On a $\vec{x} \in \mathbb{R}^n$, $\vec{g}(\vec{x}) \in \mathbb{R}^m$ et $\vec{h}(\vec{x}) \in \mathbb{R}^p$

L'approche par ε -contrainte doit aussi être appliquée plusieurs fois en faisant varier le vecteur ε, qui doit être choisit judicieusement, pour trouver un ensemble de points Pareto Optimaux.
En transformant des fonctions objectif en contraintes, elle diminue la zone réalisable par paliers. Ensuite, le processus d'optimisation trouve le point optimal sur l'objectif restant.

2.4.2 Méthodes Approchées

Certains problèmes d'optimisation demeurent cependant hors de portée des méthodes exactes. Un certain nombre de caractéristiques peuvent en en effet être problématiques, comme l'absence de convexité stricte (multimodalité), l'existence de discontinuités, une fonction non dérivable, présence de bruit, etc. Dans de tels cas, le problème d'optimisation est dit difficile, car aucune méthode exacte n'est capable de le résoudre exactement en un temps raisonnable, on devra alors faire appel à des heuristiques permettant une optimisation approchée.
Ces méthodes sont, en général, présentées sous la forme de concept d'inspiration. Plusieurs définition ont été données pour clarifier le concept de l'heuristique, parmi les quels on trouve les définitions suivantes :

Définition 9 (Heuristique selon Reeves) Une heuristique est une technique trouvant de bonnes solutions (c'est-à-dire proches de l'optimum) pour un coût de calcul raisonnable, sans pouvoir garantir l'admissibilité ou l'optimalité, ou même dans de nombreux cas, préciser la distance à l'optimum d'une solution particulière.

Métaheuristiques à base de solution unique

Parmi les heuristiques, certaines sont adaptables à un grand nombre de problèmes différents sans changements majeurs dans l'algorithme, on parle alors de métaheuristiques.

La plupart des heuristiques et des métaheuristiques utilisent des processus aléatoires comme moyens de récolter de l'information et de faire face à des problèmes comme l'explosion combinatoire. En plus de cette base stochastique, les métaheuristiques sont généralement itératives, c'est-à-dire qu'un même schéma de recherche est appliqué plusieurs fois au cours de l'optimisation, et directes, c'est-à-dire qu'elles n'utilisent pas l'information du gradient de la fonction objectif. Elles tirent en particulier leur intérêt de leur capacité à éviter les optima locaux, soit en acceptant une dégradation de la fonction objectif au cours de leur progression, soit en utilisant une population de points comme méthode de recherche (se démarquant ainsi des heuristiques de descente locale).

Définition 10 (Métaheuristique selon Osman et Laporte) Une métaheuristique est un processus itératif de génération guidant une heuristique subordonnée en combinant intelligemment différents concepts pour explorer et exploiter l'espace de recherche en utilisant des stratégies pour structurer l'information de manière à trouver efficacement des solutions proches de l'optimum.

Les métaheuristiques à base de solution unique débutent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante. En fait, la solution initiale subit une série de modifications en fonction de son voisinage. Le but de ces modifications locales est d'explorer le voisinage de la solution actuelle afin d'améliorer progressivement sa qualité au cours des différentes itérations. On explique ci aprés la recherche tabou et le recuit simulé.

Recherche tabou

Grandibleux et al [Gandibleux et al. , 1997] ont proposé une version multi objectif de la recherche taboue. La méthode utilise des fonctions pondérées scalaires dont les poids sont changés périodiquement. La modification du vecteur poids dégrade les poids des objectifs qui ont été nettement améliorés. Deux listes taboues sont utilisées. La première est une liste taboue régulière qui empêche de revenir aux solutions déjà visitées. La deuxième contient les vecteurs poids.

Hansen [Hansen, 1998] a proposé une recherche taboue basée sur l'idée de la méthode **Pareto simulated annealing**[Czyzak & Jaszkiewicz, 1998]. La méthode utilise une population de solutions explorant chacune d'elles différentes régions de l'ensemble Pareto. Le vecteur poids est utilisé dans une fonction scalaire. De plus, à chaque solution est associée une liste taboue. La dispersion des solutions est assurée par la modification de leurs vecteurs poids. Pour chaque solution, la modification du vecteur poids vise à la déplacer des autres solutions proches dans l'espace des objectifs. Hansen a appliqué cette méthode au problème du sac à dos multi objectif.

Ben Abdelaziz et Krichen [Ben Abdelaziz & Krichen, 1997] ont proposé un algorithme de recherche taboue multi-objectif conçu spécialement au problème du sac à dos multi-objectif. La méthode est guidée par des fonctions scalaires pondérées linéaires et par une relation de dominance. La relation de dominance est appliquée non seulement aux solutions mais aussi aux objets. La recherche locale est basée sur l'idée de l'insertion des objets non-dominés au sac à dos.

Le recuit simulé

Une nouvelle technique de résolution des problème d'optimisation est nommée recuit(RS) simulé, proposée en 1983 par Kirkpatrick, C.Daniel Gelatt et Mario P Vecchi [3]. Elle est testée sur plusieurs problèmes d'optimisation et prouve qu'elle possède une grande capacité pour éviter le minimum local. (RS) est une méthode basée sur la recherche locale dans laquelle chaque mouvement est accepté s'il améliore la fonction objective. Autres solutions possibles sont également acceptées selon un critère de probabilité. Cette méthode est inspirée du processus de recuit utilisé en métallurgie pour améliorer la qualité d'un solide en cherchant un état d'énergie minimum. La méthode du recuit simulé, appliquée aux problèmes d'optimisation, considère une solution initiale et cherche dans son voisinage une autre solution de façon aléatoire [57].

Métaheuristiques à base de population de solutions

Les métaheuristiques à base de population de solutions débutent la recherche avec une panoplie de solutions. Elles s'appliquent sur un ensemble de solutions afin d'en extraire la meilleure (l'optimum global) qui représentera la solution du problème traité. L'idée d'utiliser un ensemble de solutions au lieu d'une seule solution renforce la diversité de la recherche et augmente la possibilité d'émergence de solutions de bonne qualité [40]. Dans cette catégorie de méthodes on trouve les algorithmes à base de colonies de fourmis et les algorithmes évolutionnaires.

Colonies de fourmis

Les algorithmes de colonies de fourmis ont été proposés par Colorni, Dorigo et Maniezzo en 1992 [19] et appliquées la première fois au problème du voyageur de commerce. Ce sont des algorithmes itératifs à population (fourmis artificielles) où tous les individus partagent un savoir commun qui leur permet d'orienter leurs futurs choix et d'indiquer aux autres individus des choix à suivre ou à éviter. Le principe de cette métaheuristique repose sur le comportement particulier des fourmis, elles utilisent pour communiquer une substance chimique volatile particulière appelée phéromone grâce à une glande située dans leur abdomen.

Algorithmes évolutionnaires

Les algorithmes évolutionnaires s'inspire de l'évolution naturelle des êtres vivants. Ils adoptent une sorte d'évolution artificielle pour améliorer la qualité des individus de la population. En fait, ils font évoluer itérativement une population d'individus. Ces derniers représentent des solutions du problème traité. Les qualités des individus sont mesurées à chaque itération du processus de

d'évolution. En fonction de leurs qualités, les meilleurs individus seront sélectionnés pour subir des combinaisons qui permettent la production d'une nouvelle population (dite : population d'enfants). Les individus (ou une partie des individus) de la nouvelle population vont remplacer les individus de la population courante (dite : population de parents) pour construire une nouvelle génération d'individus. Le processus d'évolution d'un algorithme évolutionnaire est basé sur trois opérations principales : la sélection, le croissement et la mutation. [40]

Approches hybrides

Plusieurs approches hybrides ont été proposées dans la littérature essayant de tirer profit des avantages de chacune des méthodes utilisées ou bien pour combler certaines de ses lacunes. Plusieurs travaux ont proposé des méthodes hybrides pour résoudre des PMOs. Jaszkiewicz a proposé l'algorithme **Multi-Objective Genetic Local Search (MOGLS)** qu'il a appliqué au problème de voyageur de commerce multi-objectif [Jaszkiewicz, 2002] et au problème du sac à dos multi-objectif [Jaszkiewicz, 2002]. Cet algorithme utilise une méthode de descente pure comme opérateur de recherche locale. A chaque itération, l'algorithme construit une fonction d'utilité aléatoire ainsi qu'une population temporaire composée d'un nombre de meilleures solutions à partir des solutions générées. Ensuite, une paire de solutions sélectionnée aléatoirement recombinée. La recherche locale est appliquée à chaque solution générée.

Barichard et Hao [Barichard et Hao, 2003] ont proposé l'algorithme GTS^{MOKP} pour résoudre le problème du sac à dos multi-objectif. Cet algorithme combine une procédure génétique avec un opérateur de recherche tabou. Récemment, plusieurs algorithmes évolutionnaires qui intègre un calcul d'indicateur d'hypervolume ont été proposés dans la littérature.

3

Pareto Ant Colony Optimisation

Sommaire

3.1 Pourquoi les fourmis

Nous présenterons dans ce chapitre une nouvelles méta-heuristiques permettant de résoudre des problèmes tels que l'optimisation du portefeuille, en s'inspirant du comportement social des fourmis.

Le comportement des fourmis est un comportement collectif. Chaque fourmi a pour priorité le bien être de la communauté. Chaque individu de la colonie est à priori indépendant et n'est pas supervisé d'une manière ou d'une autre. Ce concept est appelé Hétérarchie (s'opposant à la Hiérarchie)[79], chaque individu est aidé par la communauté dans son évolution et en retour il aide au bon fonctionnement de celle-ci. La colonie est donc auto-controlé par le biais de mécanismes relativement simples à étudier.

3.2 Comportement de la fourmi

En marchant du nid à la source de nourriture et vice-versa (ce qui dans un premier temps se fait essentiellement de façon aléatoire), les fourmis déposent au passage sur le sol une substance odorante appelée phéromones. Cette substance permet ainsi donc de créer une piste chimique, sur laquelle les fourmis s'y retrouvent. En effet, d'autres fourmis peuvent détecter les phéromones grâce à des capteurs sur leurs antennes.

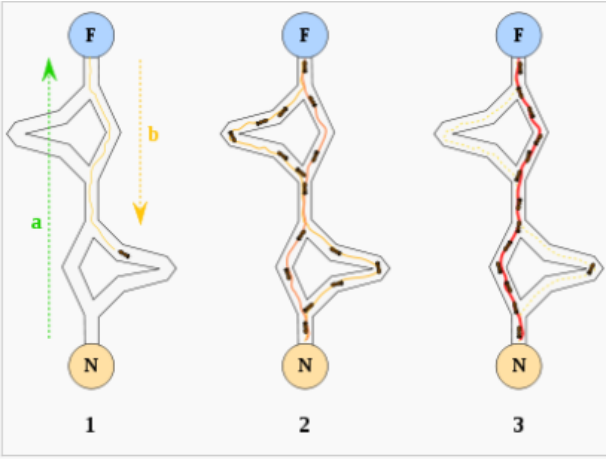
Les phéromones ont un rôle de marqueur de chemin : quand les fourmis choisissent leur chemin, elles ont tendance à choisir la piste qui porte la plus forte concentration de phéromones. Cela leur permet de retrouver le chemin vers leur nid lors du retour. D'autre part, les odeurs peuvent être utilisées par les autres fourmis pour retrouver les sources de nourritures trouvées par leurs congénères. Tandis qu'en absence de traces une fourmi se déplace aléatoirement, [21]

Ce comportement permet de trouver le chemin le plus court vers la nourriture lorsque les pistes de phéromones sont utilisées par la colonie entière. Autrement dit, lorsque plusieurs chemins marqués sont à la disposition d'une fourmi, cette dernière peut connaitre le chemin le plus court vers sa destination.

Un modèle expliquant ce comportement est le suivant :

a. une fourmi (appelée : éclaireuse) parcourt plus ou moins au hasard l'environnement autour de la colonie ;
b. si celle-ci découvre une source de nourriture, elle rentre plus ou moins directement au nid, en laissant sur son chemin une piste de phéromones ;
c. ces phéromones étant attractives, les fourmis passant à proximité vont avoir tendance à suivre, de façon plus ou moins directe, cette piste ;
d. en revenant au nid, ces mêmes fourmis vont renforcer la piste ;

e. si deux pistes sont possibles pour atteindre la même source de nourriture, celle étant la plus courte sera, dans le même temps, parcourue par plus de fourmis que la longue piste ;
f. la piste courte sera donc de plus en plus renforcée, et donc de plus en plus attractive ;
g. la longue piste, elle, finira par disparaître, les phéromones étant volatiles ;
h. à terme, l'ensemble des fourmis a donc déterminé et « choisi » la piste la plus courte.



1) la première fourmi trouve la source de nourriture (F), via un chemin quelconque (a), puis revient au nid (N) en laissant derrière elle une piste de phéromone (b). 2) les fourmis empruntent indifféremment les quatre chemins possibles, mais le renforcement de la piste rend plus attractif le chemin le plus court. 3) les fourmis empruntent le chemin le plus court, les portions longues des autres chemins perdent leur piste de phéromones.

FIGURE 3.1 – Comment les fourmis trouvent le plus court chemin

En observant une colonie de fourmis à la recherche de nourriture dans les environs du nid, on s'aperçoit qu'elle résout des problèmes tels que celui de la recherche du plus court chemin. Les fourmis résolvent des problèmes complexes par des mécanismes assez simples a modéliser. Il est ainsi assez simple de simuler leur comportement par des algorithmes.

Pour transposer ce comportement à un algorithme général d'optimisation combinatoire, on fait une analogie entre l'environnement dans lequel les fourmis cherchent de la nourriture et l'ensemble des solutions admissibles du problème(l'espace de recherche du problème), entre la quantité ou la qualité de la nourriture et la fonction objectif à optimiser et enfin entre les traces et une mémoire adaptative. Les fourmis artificielles dans les algorithmes ACO se comportent de la même manière. Elles diffèrent des fourmis naturelles dans le fait qu'elles ont une sorte de mémoire, pour assurer la génération de solutions faisables. En plus, elles ne sont pas complètement aveugles, elles ont des informations sur leur environnement.

3.3 Similarités et différences avec les fourmis réelles

Les fourmis virtuelles ont une double nature. D'une part, elles modélisent les comportements abstraits de fourmis réelles, et d'autre part, elles peuvent être enrichies par des capacités que ne possèdent pas les fourmis réelles, afin de les rendre plus efficaces que ces dernières. Nous allons maintenant synthétiser ces ressemblances et différences.

3.3.1 Points communs

Colonie d'individus coopérants. Comme pour les fourmis réelles, une colonie virtuelle est un ensemble d'entités non-synchronisés, qui se rassemblent ensemble pour trouver une "bonne" solution au problème considéré. Chaque groupe d'individus doit pouvoir trouver une solution même si elle est mauvaise.

Pistes de phéromones. Ces entités communiquent par le mécanisme des pistes de phéromone. Cette forme de communication joue un grand rôle dans le comportement des fourmis : son rôle principal est de changer la manière dont l'environnement est perçu par les fourmis, en fonction de l'historique laissé par ces phéromones.

Évaporation des phéromones. La méta-heuristique ACO comprend aussi la possibilité d'évaporation des phéromones. Ce mécanisme permet d'oublier lentement ce qui s'est passé avant. C'est ainsi qu'elle peut diriger sa recherche vers de nouvelles directions, sans être trop contrainte par ses anciennes décisions.

Recherche du plus petit chemin. Les fourmis réelles et virtuelles partagent un but commun : recherche du plus court chemin reliant un point de départ (le nid) à des sites de destination (la nourriture).

Deplacements locaux. Les vraies fourmis ne sautent pas des cases, tout comme les fourmis virtuelles. Elles se contentent de se déplacer entre sites adjacents du terrain.

Choix aléatoire lors des transitions. L lors des transitions. Lorsqu'elles sont sur un site, les fourmis réelles et virtuelles doivent décider sur quel site adjacent se déplacer. Cette prise de décision se fait au hasard et dépend de l'information locale déposée sur le site courant. Elle doit tenir compte des pistes de phéromones, mais aussi du contexte de départ (ce qui revient à prendre en considération les données du problème d'optimisation combinatoire pour une fourmi virtuelle).

3.3.2 Différences

Les fourmis virtuelles possèdent certaines caractéristiques que ne possèdent pas les fourmis réelles :

Elles vivent dans un monde non-continu. Leurs déplacements consistent en des transitions d'état.

Mémoire (état interne) de la fourmi. Les fourmis réelles ont une mémoire très limitée. Tandis que nos fourmis virtuelles mémorisent l'historique de leurs actions. Elles peuvent aussi retenir des données supplémentaires sur leurs performances.

Nature des phéromones déposées. Les fourmis réelles déposent une information physique sur la piste qu'elles parcourent, là où les fourmis virtuelles modifient des informations dans les variables d'états associées au problème. Ainsi, l'évaporation des phéromones est une simple décrémentation de la valeur des variables d'états à chaque itération.

Qualité de la solution. Les fourmis virtuelles déposent une quantité de phéromone proportionnelle à la qualité de la solution qu'elles ont découvert.

Retard dans le dépôt de phéromone. Les fourmis virtuelles peuvent mettre à jour les pistes de phéromones de façon non immédiate : souvent elles attendent d'avoir terminé la construction de leur solution. Ce choix dépend du problème considéré bien évidemment.

Capacités supplémentaires. Les fourmis virtuelles peuvent être pourvues de capacités artificielles afin d'améliorer les performances du système. Ces possibilités sont liées au problème et peuvent être :

a. L'anticipation : la fourmi étudie les états suivants pour faire son choix et non seulement l'état local.
b. Le retour en arrière : une fourmi peut revenir à un état déjà parcouru car la decision qu'elle avait prise à cet état a été mauvaise.

3.4 Optimisation par colonies de fourmis.

A l'origine, l'optimisation par colonie de fourmis a été conçue pour résoudre le problème du voyageur de commerce en proposant le premier algorithme ACO : 'Ant System' (AS) [Dorigo Gambardella, 1997]. Par la suite, un nombre considérable d'applications de ACO a été proposé telles que l'affectation quadratique [Gambardella et al. , 1999a], le routage des véhicules [Bullnheimer et al. , 1999], le problème de satisfaction de contraintes [Solnon, 2002], problème de selection de portefeuille financier [Doerner et al., 2004].

3.4.1 Algorithme de base : Ant System

Ant System est le premier algorithme fourmi proposé dans la littérature, et qui a été implémenté pour résoudre le problème de voyageur de commerce (PVC) [Dorigo, 1992].

Dans l'algorithme Ant System (AS), chaque fourmi est initialement placée sur une ville choisie aléatoirement, chacune possède une mémoire qui stocke la solution partielle qu'elle a construite auparavant. Initialement, la mémoire contient la ville de départ. Commençant à partir de cette ville, une fourmi se déplace itérativement d'une ville à une autre. Quand elle est à une ville i, une fourmi k choisit d'aller à une ville non encore visitée j avec une probabilité donnée par :

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\Sigma_{y \in N_i^k}([\tau_{iy}(t)]^{\alpha}[\eta_{iy}]^{\beta})} & si \quad j \in N_i^k \\ 0 & sinon \end{cases} \tag{3.1}$$

$\tau_{ij}(t)$ est l'intensité de la trace de phéromone dans l'arête (i, j) à l'instant t.
η_{ij} est une information heuristique à priori valable, où d_{ij} est la distance entre la ville i et la ville j ; l'idée étant d'attirer les fourmis vers les villes les plus proches.
α et β sont deux paramètres qui déterminent l'infuence relative de la trace de phéromone et de l'information heuristique.
N_i^k est le voisinage faisable de la fourmi k c'est à dire l'ensemble des villes non encore visitées par la fourmi k.

La construction de solution se termine après que chaque fourmi ait complété un tour. Ensuite, les traces de phéromone sont mises à jour. Dans AS, la mise à jour se fait, d'abord, en réduisant les traces de phéromone avec un facteur constant ρ (c'est l'évaporation de phéromone) et, ensuite, en permettant à chaque fourmi de déposer de la phéromone sur les arêtes qui appartiennent à son tour. Ainsi la formule de mise à jour de phéromone est comme suit :

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Sigma_{k=1}^{nb\ ants}\Delta\tau_{ij}^{k} \tag{3.2}$$

avec $0 < \rho < 1$ et $nbants$ est le nombre de fourmis.

Le paramètre ρ est ainsi utilisé pour éviter l'accumulation illimitée de phéromone et permet à l'algorithme d'oublier les mauvaises décisions précédemment prises. Sur les arêtes qui n'ont pas été choisis par les fourmis la force associée va décroître rapidement avec le nombre d'itérations.

$\Delta\tau_{ij}^{k}$ est la quantité de phéromone que la fourmi k dépose sur l'arête (i, j).Il est défini par :

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L^k} & si \quad (i,j) \in Tabou^k \\ 0 & sinon \end{cases} \tag{3.3}$$

où L^k est la longueur du tour généré par la fourmi k, Q une constante de l'algorithme et $Tabou^k$ est la liste des villes déjà visitées.

Avec cette formule, les arêtes du tour le plus court recevront la plus grande quantité de phéromone. En général, les arêtes qui sont utilisées par plusieurs fourmis et qui appartiennent aux tours les plus courts recevront plus de phéromone et en conséquence seront plus favorisés dans les itérations futures de l'algorithme.

3.4.2 Extensions de Ant System

Malgré les résultats encourageants trouvés pour le problème du VC, AS n'était pas compétitif avec les algorithmes de l'état de l'art du PVC. Des recherches sont alors entreprises pour l'étendre et essayer d'améliorer ses performances en contrôlant mieux l'intensification et la diversification de la recherche.

Rank-based Ant system (AS-rank)

C'est une extension proposée par Bullnheimer, Hartl, and Strauss [Bullnheimer et al. , 1999]. Cet algorithme trie les fourmis selon les longueurs de tours générés.

Après chaque phase de construction de tours, seulement les w meilleures fourmis sont autorisées à la mise à jour de phéromone. Ainsi, la r^{ime} fourmi contribue à la mise à jour de phéromone avec un poids donné par le $\max\{0, w-r\}$ tandis que la fourmi correspondant au meilleur tour global renforce la trace de phéromone avec un poids w. L'objectif de cette version est d'intensifier plus la recherche vers les meilleures solutions.

Ant Colony System

L'algorithme **Ant Colony System** (ACS) [Dorigo & Gambardella, 1997] a été introduit par Dorigo et Gambardella pour améliorer les performances du premier algorithme sur des problèmes de grandes tailles.

ACS introduit une règle qui dépend d'un paramètre $q_0 (0 \leq q_0 \leq 1)$, qui définit une balance diversification/intensification. Cette règle permet aux fourmis de favoriser plus, lors de leurs déplacements, l'arête qui contient le maximum de traces suivant q_0 , sinon la règle de transition habituelle est utilisée. Ainsi ACS permet d'intensifier plus la recherche vers les zones les plus prometteuses, i.e., qui contiennent plus de traces.

MAX - MIN Ant Sytem (MMAS)

Stützle et Hoos ont introduit l'algorithme MMAS [Stützle & Hoos, 2000] qui, pour améliorer les performances de ACO, combine une exploitation améliorée des meilleures solutions trouvées durant la recherche avec un mécanisme efficace pour éviter une stagnation prématurée de la recherche. MMAS diffère en trois aspects clé de AS :

- Pour exploiter les meilleures solutions trouvées durant une itération ou durant l'exécution de l'algorithme, après chaque itération, une seule fourmi ajoute de la phéromone. Cette fourmi peut être celle qui a trouvé la meilleure solution depuis le début de l'exécution, ou bien celle durant le dernier cycle
- Pour éviter une stagnation de la recherche, les valeurs possibles de la trace de phéromone sur chaque composant de solution sont limitées à l'intervalle $[\tau_{min}, \tau_{max}]$
- De plus, les traces de phéromone sont initialisées à τ_{max} pour assurer une exploration élevée de l'espace des solutions au début de l'algorithme.

Contrôle de l'intensification et de la diversification

Un point critique lors de l'application de l'algorithme ACO, est de trouver un équilibre entre l'exploitation de l'expérience de recherche acquise par les fourmis et l'exploration de nouvelles zones de l'espace de recherche non encore visitées. ACO possède une bonne manières pour accomplir une telle balance, essentiellement par la gestion des traces de phéromone. En effet, les traces de phéromone déterminent dans quelles parties de l'espace de recherche les solutions construites auparavant sont localisées.

Une manière de mettre à jour la phéromone, et pour exploiter l'expérience de recherche acquise par les fourmis, est de déposer une quantité de phéromone fonction de la qualité de la solution trouvée par chaque fourmi. Ainsi, les zones correspondant aux meilleures solutions recevront une quantité de phéromone plus élevée et seront plus sollicitées par les fourmis durant leurs déplacements.

Pour contrôler la balance entre l'exploitation de l'espace de recherche et l'exploration de nouvelles zones, ACO dispose des paramètres α et ρ pour gérer l'importance relatives des traces de phéromone.

α détermine le poids des traces de phéromone dans le calcul des probabilités de transition, et ρ détermine l'évaporation de phéromone. Ces deux paramètres ont une influence sur le comportement exploratoire ; pour favoriser la stratégie d'exploration, on peut diminuer le coefficient α , ainsi les fourmis deviennent moins sensibles aux phéromones lors de leurs déplacements et peuvent visiter des zones non explorées, ou diminuer ρ, ainsi la phéromone s'évapore plus lentement et l'intensité des traces de phéromone devient similaire sur les arêtes et donc les fourmis deviennent moins influencées par la phéromone.

Pour favoriser la stratégie d'exploitation, on augmente les valeurs respectives de α et/ou de ρ, ainsi les fourmis seront plus guidées par la phéromone vers des zones prometteuses de l'espace de recherche. Après un certain nombre d'itérations, toutes les fourmis vont converger vers un ensemble de bonnes solutions. Ainsi, on favorise une convergence plus rapide de l'algorithme.

Toutefois, il faut faire attention et éviter une intensification trop forte dans les zones qui apparaissent prometteuses de l'espace de recherche car cela peut causer une situation de stagnation :

une situation dans laquelle toutes les fourmis génèrent la même solution.

Pour éviter une situation pareille de stagnation, une autre solution serait de maintenir un niveau raisonnable d'exploration de l'espace de recherche. Par exemple, dans ACS les fourmis utilisent une règle de mise à jour locale de phéromone durant la construction de solution pour rendre le chemin qu'elles viennent de prendre moins désirable pour les futures fourmis et ainsi, diversifier la recherche. MMAS introduit une limite inférieure pour l'intensité des traces de phéromone pour garantir toujours la présence d'un niveau minimal d'exploration. MMAS utilise aussi une réinitialisation des traces de phéromone, qui est une manière de renforcer l'exploration de l'espace de recherche.

3.4.3 Métaheuristique de Pareto Ant Colony Optimization

Dans les métaheuristique d'optimisation par colonies de fourmis ACO [Dorigo et Di Caro, 1999, Dorigo et Stüzle, 2004], les fourmis se déplacent en appliquant une politique de décision stochastique locale qui se base sur l'utilisation des traces de phéromone et d'une information heuristique spécifique au problème traité. Des coefficients α et β permettent de contrôler l'importance relative des deux éléments. Chaque fourmi possède une forme de mémoire afin d'obliger celle-ci à former une solution admissible.

En se déplaçant, les fourmis construisent incrémentalement des solutions au problème d'optimisation. Une fois qu'une fourmi a construit une solution, ou lorsque la solution est en cours de construction, la fourmi évalue la solution (partielle) et dépose la phéromone dans le composant ou la connexion qu'elle a utilisé. Cette information de phéromone dirigera la recherche des futures fourmis. [3]

L'intensité de ces traces décroît dans le temps par un facteur constant appelé coefficient d'évaporation. D'un point de vue pratique, l'évaporation de phéromone est nécessaire pour éviter une convergence prématurée de l'algorithme vers une région sous-optimale. Ce processus implémente une forme utile d'oubli favorisant l'exploration de nouvelles aires de recherche.

Récemment plusieurs travaux ont proposé des algorithmes basés sur l'optimisation par colonies de fourmis ACO pour résoudre des Problèmes multiobjectifs, appelés dans la littérature algorithmes MOACO.

Pareto Ant Colony Optimization (P-ACO)

Pareto Ant Colony Optimization (P-ACO) a été proposé par Doerner [Doerner.2004]. Il a été initialement appliqué au problème de sélection de portefeuille multi-objectif. Il suit le schéma général de l'algorithme Ant System (AS).

Dans la phase initiale de cet algorithme, on commence d'abord par générer Γ fourmis sachant que chaque fourmi a un portefeuille vide $y = (0, \ldots, 0)$ qu'elle doit remplir au fur et mesure.

A chaque cycle de l'algorithme, chaque fourmi calcule un ensemble de poids $P = (P_1, P_2, \ldots, P_K)$ et l'utilise pour combiner les traces de phéromone et l'information heuristique afin de construire

un portefeuille faisable x, en appliquant une règle basée sur l'information heuristique η_i et sur la structure phéromone τ_i. Une fois le portefeuille est construit,l'algorithme test la faisabilité et l'efficacité c'est-à-dire : si le portefeuille construit est faisable et efficace il sera stocké dans un ensemble externe.

A la fin, la mise à jour globale de la phéromone est réalisée en utilisant deux fourmis, la meilleure et la deuxième meilleure solution générée dans le cycle courant pour chaque objectif K.

3.4.4 La règle de décision

Pour chaque objectif k, les structures phéromones sont stockées dans un vecteur de dimension N et pour chaque projet candidat, une valeur agrégée de l'attractivité est calculée.

Soit η_i l'attractivité, τ_i^k la structure phéromone et $\Omega(x)$ l'ensemble de tous les portefeuilles faisables tel que :

$$\Omega(x) = \{i \in N, \eta_i(x) > 0,\ x_i = 0\} \qquad (3.4)$$

Les fourmis artificielles sélectionnent un portefeuille faisable $i \in \Omega(x)$ afin qu'il soit rajouté a l'ensemble des solutions et ce selon la règle suivante :

$$i = \begin{cases} \arg\max\limits_{i \in \Omega(x)} \{ [\sum\limits_{k=1}^{K} (p_k \tau_i^k)]^{\alpha} [\eta_i(x)]^{\beta} \} & if\ q \leq q_0 \\ \\ j & ailleurs \end{cases} \qquad (3.5)$$

Où q : variable aléatoire qui suit une loi uniforme sur l'intervalle [0,1]. q_0 : Un paramètre compris entre 0 et 1 représente la probabilité que la solution choisie a les plus grandes valeurs agrégées de l'attractivité et la phéromone. Quant a la variable aléatoire j elle est sélectionnée selon la distribution de probabilité suivante :

$$p_i(x) = \frac{\left[\sum\limits_{k=1}^{K} (p_k \tau_i^k) \right]^{\alpha} [\eta_i(x)]^{\beta}}{\sum\limits_{h \in \Omega(x)} \left[\left[\sum\limits_{k=1}^{K} (p_k \tau_h^k) \right]^{\alpha} \left[\eta_h(x)^{\beta} \right] \right]} \qquad (3.6)$$

On remarque que cette expression est donnée par les α et β qui représentent respectivement le poids des traces de phéromone dans et la visibilité.

3.4.5 La mise à jour de phéromone

Une mise à jour locale de phéromone est effectuée une fois une fourmi artificielle ajoute un projet à un portefeuille. Quand une fourmi choisit un projet i, la quantité de phéromone sur l'élément τ_i^k du vecteur phéromone diminue pour chaque k objectif. La règle de mise à jour locale de phéromone pour ces éléments est donnée comme suit :

$$\tau_i^k = (1-\rho)\tau_i^k + \rho\tau_0 \qquad (3.7)$$

Où τ_0 est la valeur initiale des traces des phéromones et ρ le taux d'évaporation.

Après la construction des solutions par chaque fourmi, La mise à jour globale de phéromone est effectuée et les tests de faisabilité et d'efficacité sont déterminés.

La mise à jour globale de phéromone se fait en utilisant deux fourmis, la meilleure et la deuxième-meilleure solution générée dans le cycle courant pour chaque objectif k comme suit :

$$\tau_i^k = (1-\rho)\ \tau_i^k + \rho\ \Delta\tau_{ij}^k \qquad (3.8)$$

Les tests sur la stratégie de la mise à jour de la phéromone, ont montré que l'utilisation de la meilleure et deuxième meilleure fourmi avec des quantités de phéromone de 10 pour la meilleure et 5 pour la deuxième, donne de bons résultats.

$$\Delta\tau_{ij}^k = \begin{cases} 10 & if\ x_{meil\ sol,i} = 1 \\ 0 & sinon \end{cases} \qquad (3.9)$$

Une fois cette mise à jour est effectuée, tout en utilisant la meilleure solution, une autre mise à jour similaire à la première sera effectuée mais cette fois en utilisant la deuxième meilleure solution et non pas la première et donnée comme suit :

$$\Delta\tau_{ij}^k = \begin{cases} 5 & if\ x_{deux\ meil\ sol,i} = 1 \\ 0 & sinon \end{cases} \qquad (3.10)$$

3.4.6 L'influence des paramètres α et β sur la résolution

Quand on résout un problème d'optimisation combinatoire avec une approche heuristique, il s'agit de trouver un bon compromis entre deux objectifs relativement duaux : d'une part il s'agit d'intensifier la recherche autour des zones de l'espace de recherche les plus prometteuses, qui sont généralement proches des meilleures solutions trouvées ; d'autre part il s'agit de diversifier la recherche et favoriser l'exploration afin de découvrir de nouvelles et si possible meilleures zones de l'espace de recherche. Le comportement des fourmis par rapport à cette dualité entre

intensification et diversification peut être influencé en modifiant les valeurs des paramètres.

En particulier, la diversification peut être augmentée soit en diminuant la valeur du poids du facteur phéromone α (de sorte que les fourmis deviennent moins sensibles aux traces phéromonales), soit en diminuant le taux d'évaporation ρ (de sorte que la phéromone s'évapore plus doucement et les écarts d'une trace à l'autre évoluent plus doucement). Lorsque l'on augmente ainsi la capacité exploratoire des fourmis, on trouve généralement de meilleures solutions, mais en contrepartie ces solutions sont plus longues à trouver.

Conclusion

Dans ce chapitre, nous avons fait une étude récapitulative menée sur les méthodes de résolution des problèmes d'optimisation multiobjectif, tout en montrant la manière de définir un problème pareil, ses contraintes, ses objectifs, tout en respectant le concept de compromis et les frontières de Pareto. Nous avons parlé de méthodes utilisées dans le domaine de l'optimisation multiobjectif à savoir exactes et métaheuristiques.

4

Algorithmes Evolutionnaires

Sommaire

4.1 Bref Historique sur les Algorithmes Evolutionnaires (AE)

Quelques chercheurs des années 1950 ont tenté d'en adapter le principe de l'évolution naturelle à l'ingénierie. Mais la faible puissance des machines de l'époque et les connaissances de la génétique naturelle n'a pas permis d'obtenir des résultats satisfaisants. Plus tard dans les années 1960 et 1970 trois grandes écoles à travers le monde utilisent les Algorithmes Evolutionnaires (AE) de façons différentes. Ces recherches ont mené à quatre grandes familles d'AE : les Algorithmes Génétiques (AG), les Stratégies d'Evolution (SE). la Programmation Evolutionnaire (PE) et la Programmation Génétique (PG).

Ceux sont des méthodes d'optimisation stochastiques basées sur les principes de l'évolution biologique naturelle. Ils recommandent les AG développés par John Holland et les SE développées par Rechenberg. Ils paraissent concilier au mieux généralité, puissance et facilité de programmation. Les chercheurs étudient le concept de l'évolution naturelle dans le but de son utilisation dans les méthodes d'optimisation en engineering.

Les Stratégies d'Evolution (SE) par Rechenberg (1965) à Berlin, Germany

C'est au milieu des années 1960 qu'ont commencé les travaux autour des SE. Elles sont conçues principalement pour résoudre les problèmes sur l'optimisation paramétrique (liste de paramètres) à valeurs réelles. Elles sont développées par la suite par H. P. Schwefel en 1981. Ce sont des algorithmes de recherche et d'optimisation évoluant suivant les trois opérateurs de sélection pour la reproduction, de recombinaison (faible croisement) et de mutation (fortement utilisée) qu'on développera par la suite.

Les Algorithmes Génétiques par John Holland (1960) Michigan University, USA

Peu de temps après, J. Holland s'intéresse à ce qu'allait devenir les AG (théorie des processus d'adaptation à leur environnement [50] naturels et comment ces mécanismes puissent être instrumenté dans le monde des sciences informatiques (sur ordinateur). C'est la technique principale la plus populaire pour développer la théorie et les procédures nécessaires à la création de programmes et de machines générales avec une capacité illimitée d'adaptation à des environnements arbitraires. Ses travaux ont trouvé un premier aboutissement en 1975 pour réaliser le rôle fondamental de la sélection non naturelle, la survie artificielle du plus adapté dans tout programme machine que l'on veut concevoir [50]. La notion d'AG a été considérablement développée pendant les années 1980 par David Goldberg pour publier son ouvrage [44], qui a largement contribué à leur développement en 1994. Dans les AG, les solutions potentielles sont représentées par des chaînes codées binaires [50] ou réelles [44], [83] dont les algorithmes évoluent suivants les trois principaux opérateurs mais subissant fortement l'opérateur croisement et faiblement la mutation pour la diversité. Ce sont des algorithmes d'exploration et d'optimisation définis par deux principes : Survie des individus les mieux adaptés (sélection naturelle) et recombinaison génétique (hérédité).

La Programmation Evolutionnaire de L. J. Fogel (1966), Californie, USA

Fogel [62] développe une technique évolutionnaire pour faire évoluer une population d'une machine à états finis par mutation aléatoire de leurs diagrammes de transition et sélection du meilleur individu (Fitness). Les contrôleurs se présentent sous forme de logiciels informatiques puis exprimés dans le formalisme d'un programme informatique et manipulés par l'AE. C'est le cas des automates à états finis qui sont des programmes représentables sous la forme de graphes de fluence dont les nœuds sont les états et les arêtes les transitions conditionnelles auxquelles sont associés une action. C'est l'évolution de programmes.

La Programmation Génétique de John Koza (1980, 1989, 1990, 1992), Californie, USA.

Plus tard, une autre classe d'AE vit le jour, la PG. C'était un sous groupe des AG. C'est la programmation sur ordinateur par l'intermédiaire de la sélection naturelle (1992). Il s'agit de développer des structures plus complexes telles que des programmes plus complexes sur ordinateur. Ce type de programme peut être représenté sous forme d'arbres dont les nœuds sont les instructions, la racine est la fonction principale et les feuilles en sont les paramètres. Koza en 1992 [59] a fait évoluer des programmes en utilisant un formalisme fonctionnel basé sur les arbres.

4.2 Principes généraux

Actuellement, la distinction entre les quatre approches est de plus en plus floue, le génotype étant souvent constitué d'un mélange de structures complexes (liste réelle de paramètres en SE, chaîne binaire en AG, graphe de fluence en PE et arbres en PG,...). La différence est essentiellement d'ordre historique. On les regroupe ainsi sous le nom d'AE qui sont des algorithmes de recherche stochastiques et des méthodes d'optimisation d'ordre général. Ces quatre techniques reposent sur le même principe. Elles ne diffèrent que sur les détails d'implantation des opérateurs et sur les procédures de sélection et de remplacement de la population.

Les AE constituent une méthode d'exploration automatique d'un espace de recherche potentiellement très vaste. Ces techniques travaillent sur une population, initialement aléatoire, d'individus représentant chacun une solution au problème traité. Une telle répartition de ces solutions dans l'espace de recherche limite les risques de convergence prématurée vers les extremums locaux. Les AE constituent ainsi une méthode de recherche guidée par les performances.

Malgré que leur objectif soit différent à l'origine, ils sont récemment utilisés pour résoudre des problèmes complexes d'optimisations. Conçus initialement pour optimiser des paramètres d'un contrôleur dont la structure est donnée, ils peuvent intervenir dès la phase de conception pour obtenir automatiquement cette structure de contrôle. Les AE utilisent des modèles de calcul de procédés évolutionnaires dans la réalisation et l'implantation pour résoudre des problèmes à base des ordinateurs.

Une définition générale et la classification de ces techniques évolutionnaires sont données par Bäch en 1996. Il définit un AE comme étant un algorithme de recherche et d'optimisation inspirée du procédé de l'évolution naturelle, qui maintient une population de structures évoluant suivant des règles de sélection avec d'autres opérateurs tels que la recombinaison et la mutation. Ces opérateurs jouent un rôle prépondérant dans la réussite d'un AG qui réalise la recherche de la solution optimale sur une population de solutions possibles, de façon parallèle.

La structure de tous les algorithmes basés sur l'évolution est décrite par le logigramme suivant :

Algorithme 1 Algorithmes basé sur l'évolution.

1 : **Initialisation** $t = 0$;
2 : Créer une population initiale $Pop\ (t)$;
3 : Evaluer $Pop\ (t)$;

4 : Tant que (critère non terminé) Do ;
5 : t = t+1 ;
6 : Sélectionner des individus pour la reproduction $Pop\ (t)$ à partir de $Pop\ (t-1)$;
7 : Altérer Pop (t) par les opérateurs de croisement et de mutation ;
8 : Evaluer nouvelle population $Pop\ (t)$;

Essayons de détailler les algorithmes génétiques qui sont les plus connus et paraissent concilier au mieux généralité, puissance et facilité de programmation des AE.

4.3 Algorithmes génétiques

Les AG s'inspirent des méthodes fondées sur la théorie de l'évolution et de la génétique moderne. Ils agissent sur une population d'individus, tout d'abord définis et codés sous forme de chaînes de caractères (binaires, réelles, codage de Gray,... avec passage d'une à l'autre) assujettis à une sélection Darwinienne : la survie et la reproduction des individus les mieux adaptés survivent à leur environnement.

Chaque individu représente un point de recherche dans l'espace des solutions, à qui on associe une valeur de fonction coût (fitness) dont on veut obtenir la valeur maximale. A partir d'une population initiale créée aléatoirement, les AE génèrent de nouveaux individus plus performants en effectuant des opérations génétiques. De nouvelles approches ont été développées à partir de la version originale.

4.3.1 Organisation d'un AG

L'AG est organisé en cinq niveaux :

Il manipule une population de taille *N* dans le but d'obtenir une solution ou un ensemble de solutions les plus adaptées au problème ; la fonction objectif dont on cherche l'optimum et qu'on définira par la suite.

La population (Pop) est composée de *N* individus ou variables dont chacun représente le codage d'une solution potentielle au problème à résoudre donnée sous forme d'une chaîne de caractères ou chromosomes. Chaque chromosome est constitué d'informations élémentaires appelées gènes qui représente une des caractéristiques de l'individu.

Chaque gène peut prendre différentes valeurs appelées allèles, dans une représentation de codage donné. Du point de vue informatique, on utilise dans les algorithmes un codage binaire où les seuls allèles possibles sont 0 et 1, c'est à dire qu'un chromosome étant constitué de gènes associés mis bout à bout pour former un individu dans l'espace de recherche.

On aboutit à une structure présentant cinq niveaux d'organisation comme suit :

Population/Individus/Chromosomes/Gènes/Bits.

Chaque dispositif est représenté par un individu doté d'un génotype constitué d'un ou de plusieurs chromosomes.

Génotype : C'est une représentation individuelle, généralement binaire (chaîne de bits). Cette information est manipulée par l'AG.

Phénotype : C'est la valeur réelle de chaque individu après décodage (le ou les paramètres décodés).

Donc un simple AG consiste en un plan adaptatif pour sélectionner des GENOTYPES de succès (meilleures chaînes de bits) pour créer des descendants pour une nouvelle population par l'intermédiaire d'un ensemble d'opérateurs génétiques prêtés de la nature. Le fonctionnement d'un AG multiobjectifs a une structure similaire à celle présentée dans l'organigramme de la figure précédente. Dans un AG, une population est composée d'un ensemble d'individus, tous différents, de longueur finie. Chacun peut être une solution candidate au problème traité.

Deux niveaux de représentation différente peuvent être considérés pour chaque individu. Chaque solution est représentée par un génotype et s'exprime sous la forme d'un phénotype.

C'est le système dont on cherche à optimiser ou concevoir le comportement. Il est donc nécessaire la mise au point d'une Fitness permettant de distinguer les génotypes les plus performants. Ces derniers ont une probabilité plus importante de léguer leurs génotypes à leurs descendants éliminant ainsi les très mauvaises solutions pour favoriser les plus performantes. Après élimination, les gènes des solutions sélectionnées sont combinés pour obtenir une nouvelle population

qui doit être mieux adaptée au problème considéré.

Les règles régissant cette transmission de gènes sont décrites sous la forme d'opérateurs génétiques.

Un AG simple travaille sur une population initiale aléatoire de chromosomes de longueur finie. A chaque itération (génération t), la population évolue pour construire une nouvelle génération par application des principaux opérateurs génétiques de sélection et de reproduction : le croisement et la mutation.

La sélection décrit la manière dont les individus sont choisis pour la reproduction. La reproduction représente la phase de l'évolution génétique par application des opérateurs de croisement et de mutation :

Le croisement correspond à la méthode utilisée pour mélanger les génotypes des parents.

La mutation décrit la fréquence affectant le génotype et la nature du changement lors de la transmission des gènes.

Les individus sont choisis sur la base de leur mesure de FITNESS pour jouer le rôle de parents des descendants qui constituent la nouvelle génération. Ce processus est répété jusqu'à ce que l'on obtienne une solution jugée satisfaisante.

Les AG font partie des approches évolutionnaires, très bien adaptés au traitement de problèmes d'optimisation multiobjectifs et suivent le même principe de fonctionnement :

- Initialisation
- Evaluation
- Sélection
- Croisement
- Mutation
- Mise à jour pour la construction de la nouvelle génération en favorisant les individus possédant une fitness la plus élevée
- Boucle de retour à la phase 2 (Evaluation)
- Recommencer la boucle du programme jusqu'à ce que les conditions d'arrêt soient remplies (critère de convergence).

4.3.2 Caractéristiques des algorithmes génétiques

Les algorithmes génétiques, en tant qu'approches évolutionnaires de problèmes, se caractérisent par cinq aspects différents par rapport aux méthodes conventionnelles d'optimisation :

a. **Le codage** : Les AG manipulent le codage de l'ensemble des paramètres (variables) plutôt

que les paramètres eux mêmes utilisés dans les autres méthodes d'optimisation du problème à traiter (sauf codage réel). Ils sont codés sous forme de chaînes de longueur finie pour former une population binaire, la plus courante, qui s'appuie sur la représentation $0, 1$ comme illustré sur la figure suivante :

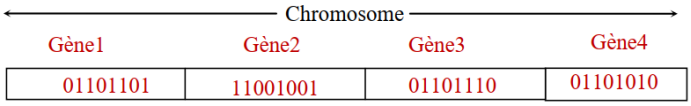


FIGURE 4.1 – Codage binaitre des paramètres

D'autres chercheurs utilisent le codage réel et ont même effectué une comparaison pour une population initiale générée par les AG en binaire et celle en réelle pour montrer que cette dernière donne de meilleurs résultats d'après leurs problèmes d'optimisation de fonction numériques à résoudre. Finalement, il n'existe pas de consensus sur la représentation idéale des individus. Mais la représentation binaire crée des problèmes pour l'optimisation des systèmes de grandes dimensions à haute précision numérique. Dans ces conditions, la taille des chromosomes binaires devient énorme amenant à de faibles performances.

b. **l'espace de recherche** : La résolution d'un problème d'optimisation consiste à explorer un espace de recherche pour maximiser (ou minimiser) une fonction. L'espace de recherche de solutions d'un AG, dont seule la performance ponctuelle est utilisée, est une population de chaînes (non pas un seul) générée aléatoirement. Il est nécessairement fini et comprend à la fois des candidats très performants alors que d'autres le sont moins. Pour des raisons technologiques et informatiques, les intervalles de définition des variables sont limités. A priori, on peut dire qu'une méthode d'optimisation déterministe est adaptée à un espace de recherche petit mais complexe et qu'un espace de recherche grand nécessite une méthode stochastique

c. **La fonction d'évaluation (fitness)** : Les AG sont des méthodes d'optimisation d'ordre zéro. Ils n'utilisent que les valeurs de la fonction étudiée, non pas ses dérivées ou autres connaissances auxiliaires (continuité, dérivabilité. . .). L'évaluation des individus, leur qualité ou leur coût de l'espace de recherche s'effectue ainsi à l'aide d'une fonction appelée fitness ou valeur sélective permettant d'associer une valeur à chaque individu. Ces valeurs serviront au processus de sélection des candidats aptes à la reproduction. Seule la capacité à calculer une valeur ponctuelle de la fonction d'évaluation est nécessaire. Le calcul de la qualité est essentiel aux AG. Le résultat en valeur numérique positive (habituellement réelle) fournit par la fitness représente la qualité d'un individu qui va permettre sa chance de sélection ou non. L'utilisation des AG est une technique d'optimisation qui cherche la qualité maximale ; donc l'optimisation de la fonction coût.

d. **La fonction objectif** : Les AG utilisent des règles de transition probabilistes et non déterministes pour la fonction d'évaluation et la génération d'une population d'individus. La fonction objectif représente comment les individus évoluent dans le problème réel (l'ap-

plication) à résoudre. Elle donne les performances d'un individu. Cette fonction peut être inférieure ou égale , ou à caractère minimal si l'individu est optimal.

4.3.3 Procédé de sélection des individus pour la reproduction

La sélection pour la reproduction est un procédé dans lequel chaque chaîne est copiée en fonction des valeurs de la fonction à optimiser nommée fonction d'adaptation qui peut être envisagée comme une mesure de profit, d'utilité ou de qualité, que l'on souhaite maximiser. Son application est pour la création d'une nouvelle population par l'utilisation d'une méthode de sélection appropriée. Le plus important est de permettre aux individus d'une population la survie (se reproduire) ou de mourir. Généralement, la possibilité de survie d'un individu sera reliée à son efficacité relative au sein de la population. On distingue deux types de sélection : La sélection déterministe et la sélection stochastique.

Sélection déterministe (valeur exacte)

La sélection déterministe est l'un des AE qui considère que la sélection ne doit pas être laissée au hasard, de l'ouvrage de Goldberg [44]en 1994. Elle consiste à sélectionner toujours les Tgénérer la suivante. C'est une sélection par troncature « truncation selection », qui suppose un tri de l'ensemble de la population. On parle alors d'élitisme généralisé à tout le processus de sélection. C'est la plus radicale qui consiste à ne conserver que les meilleurs individus. Les risques de perte de la diversité sont importants avec cette stratégie.

Sélection stochastique (avec une certaine probabilité)

Dans cette stratégie plus souple que la précédente, on favorise toujours les meilleurs individus mais de manière stochastique (probabiliste), ce qui laisse une chance aux individus moins performants. Copier des chaînes en fonction des valeurs de leur fonction d'adaptation revient à donner aux chaînes dont la valeur est plus grande une probabilité plus élevée de contribuer à la génération suivante, en créant au moins un descendant. Il se peut que le meilleur individu ne soit pas sélectionné au dépend du plus faible et qu'aucun des enfants ne soit au niveau du meilleur parent.

La sélection peut être mise en œuvre sous forme algorithmique de différentes manières. Sa plus simple forme a été décrite par Goldberg en 1989 [42], en utilisant la sélection proportionnelle par roulette biaisée « RWS : Roulette Wheel Selection » permettant de créer une nouvelle population à partir de l'ancienne. On trouve aussi s´election par le rang, s´election par tournoi,...ect

Sélection proportionnelle par roulette (RWS)

La sélection proportionnelle « Proportional Selection ou Roulette Wheel Selection : RWS » est la plus ancienne de toutes les méthodes de sélection, qui consiste à affecter à chaque individu courant dans la population, une probabilité d'être sélectionnée PSel.i, occupant un secteur dans

la roulette subdivisée, proportionnelle à sa fitness $F(x_i)$ représentée sur la figure ci après, Il s'agit de quatre chaînes codées binaires (4 chromosomes) de cinq bit chacune tirées aléatoirement (20 tirages), de l'exemple de Goldberg pour trouver le maximum de la fonction $f(x) = x^2$ dans un espace de 32 valeurs possibles de x, sur l'intervalle $[0, 31]$.

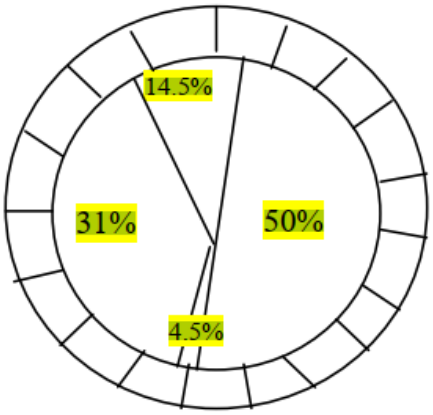


FIGURE 4.2 – Roulette biaisée.

On fait tourner cette roulette autant de fois qu'il y' a d'individus et on obtient en final la nouvelle population.

Chaque individu de valeur phénotypique x_i possède donc une probabilité PSel.i d'être conservée proportionnelle à ses performances. Il sera réintroduit dans la nouvelle population de taille N avec une probabilité de sélection proportionnelle à $F(x_i)$

$$p_i = \frac{F(x_i}{\sum_{j=1}^{N} F_j} \qquad (4.1)$$

Lors de la sélection, les individus sont sélectionnés aléatoirement en respectant les probabilités Pseli associées pour former la population de la nouvelle génération. Ceci s'effectue par le calcul des probabilités cumulatives qi définie par :

$$q_i = \sum_{\substack{i \\ j=1}} p_j \qquad (4.2)$$

Pour cela, on fait tourner N fois la roulette pour générer des nombres réels r sur l'intervalle $[0, 1]$ qu'on notera r_1, r_2,...,r_N. L'individu v_i est sélectionné lorsque : $q_i < r < q_{i+1}$

Il est clair que ce modèle se fait avec remise, où les individus de grande fitness ont plus de chance d'être sélectionnés plusieurs fois et conservés, mais ceux de performance plus faible peuvent également rester d'une génération à l'autre aidant à la préservation de la diversité, tandis que les plus mauvais risquent de mourir. Ils sont alors ajoutés dans le groupe qui constitue

l'ébauche de la nouvelle génération, qui est ensuite soumise à d'autres opérateurs génétiques.

Sélection par Tournois (Tournament Selection)
Dans la sélection par tournois, la compétition aura lieu entre les individus choisis aléatoirement d'une sous population de taille N. Pour sélectionner un individu, on en tire deux ou N uniformément dans la population et on sélectionne d'une manière déterministe celui de meilleur performance des deux (ou plus) qui pourra encore participer à un tournoi pour la reproduction de la nouvelle population.

Au cours d'une génération t, on organise autant de tournois par paire (couple) qu'il y'a d'individus N à repêcher jusqu'à ce que la nouvelle population soit complète.

Sélection par rang

La sélection par roulette présente des inconvénients lorsque la valeur d'´évaluation des individus varie énormément. En effet, on risquerait d'atteindre une situation de stagnation de l'´évolution. Imaginons la moitié de la roulette est allouée à l'individu qui a la meilleure évaluation, alors les autres individus auront une probabilité très faible d'être sélectionnés.

La sélection par rang trie d'abord la population par évaluation. Ensuite, chaque individu se voit associé un rang en fonction de sa position. Ainsi le plus mauvais individu aura le rang 1, le suivant 2, et ainsi de suite jusqu'au meilleur individu qui aura le rang N, pour une population de N individus.

4.3.4 Opérateurs de reproduction pour la génération de nouveaux individus

Les nouveaux individus sont des copies d'individus parents sélectionnés en fonction de leurs performances. Une fois certains individus sélectionnés par la méthode choisie, deux types d'opérateurs leurs sont appliqués : Le croisement et la mutation dont le résultat de leur application dépend de tirages aléatoires.

Le croisement pour la reproduction des individus de sélection

Le croisement est l'opérateur principal de recherche d'un AG, car il permet d'améliorer les performances de la population en manipulant la structure des génotypes des individus. Il est communément inclus avec une haute probabilité.

Il existe plusieurs méthodes pour effectuer le croisement à la fois au niveau génotype et phénotype, Goldberg 1989 [42], Wright 1991 [83], Michalewicz [**?**]et Dejong 1992 [27].

Le croisement combine les génotypes de deux individus pour obtenir deux nouveaux auxquels on applique la fitness du problème considéré pour trouver le plus performant. Il consiste

à échanger des gènes entre deux individus « parents » arbitrairement choisis dans la population produisant ainsi de nouveaux individus « enfants » qui possèdent certaines caractéristiques génétiques de leurs parents respectifs. Avec cet opérateur, les génotypes sont vus comme une chaîne de nombres binaires, contrairement aux phénotypes à valeurs réelles. Parmi les méthodes utilisées, il peut y avoir 1, 2 ou un nombre quelconque de points de croisements l compris entre 1 et $s-1$ qui s'effectue avec une certaine probabilité Pc qui varie généralement de $0,6$ à 0.95.

Croisement simple (ou à point unique)

Le croisement simple consiste à choisir au hasard (tirage aléatoire) un seul point de coupure identique sur les deux génotypes parents de longueur s pour échanger les fragments de chromosomes situés après le point de coupure pour produire deux descendants. L'opérateur de croisement le plus simple, dont le lieu de coupure (site de croisement l=5) est sélectionné aléatoirement, est représenté en figure ci- dessous :

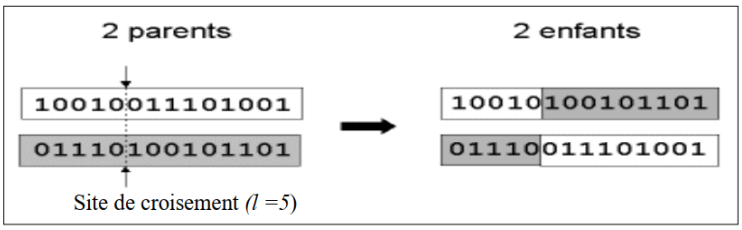


FIGURE 4.3 – Croisement simple sur une chaine binaire

Croisement double et croisement multiple)

Pour le croisement double (ou en deux points), on choisit aléatoirement deux points de croisement pour échanger les fragments. et pour le croisement multiple (ou en k points) C'est la généralisation à k points de coupure des deux précédentes où plusieurs points sont sélectionnés et l'échange se fait sur les différentes parties des séquences cernées par ces points [27]. Le cas limite est le croisement binaire uniforme qui procède à l'échange de chaque élément selon une probabilité Pc.

Croisement arithmétique simple Dans ce mode de croisement arithmétique simple [44], les valeurs des gènes sélectionnés pour la génération $t+1$ par l'opérateur de croisement appliqué aux chromosomes parents A_1^t et A_2^t de la génération t sont obtenues par les expressions ci dessous :

$$\begin{aligned} A_1^{t+1} &= \alpha A_1^t + \beta A_2^t \\ A_2^{t+1} &= \alpha A_2^t + \beta A_1^t \end{aligned} \qquad (4.3)$$

Où les constantes α et β sont des nombres obtenus aléatoirement : $\alpha, \beta \in [0,1]$. Cet opérateur est appliqué à certains éléments (gènes) sélectionnés.

Mutation appliquée sur les nouveaux individus

Une mutation est l'inversion d'un bit aléatoire dans un chromosome. Elle jouent le rôle de bruit et assurent en outre une recherche aussi bien globale que locale, selon le poids et le nombre des bits mutés. Plus un individu a été muté, plus la recherche s'éloignera du voisinage local de ce même individu non muté.

L' opérateur de mutation [64] est un opérateur qui permet à un algorithme génétique d'atteindre tous les points de l'espace d'état d'une manière susceptible, sans les parcourir tous dans le processus de résolution, ce qui permet la convergence des algorithmes génétiques vers l'optimum global.Cette opération consiste à remplacer aléatoirement un gène dans le chromosome par une valeur aléatoire. Celle-ci peut être choisie dans le voisinage de la valeur initiale [34].

En général, on décide de muter une solution avec une probabilité assez faible. Le but de la mutation est d'introduire un élément de diversification et d'innovation. Il existe de nombreuses méthodes simulant une mutation. Souvent la probabilité de mutation P_m par bit et par génération est fixée entre 0, 001 et 0, 01. On peut prendre également une valeur $P_m = 1$. La mutation peut créer des chromosomes difficiles à avoir avec le croisement. Elle correspond à une petite perturbation de la solution. On cite les quatre variantes suivantes :

Modifier la valeur d'un gêne : Le gêne de position 3 est modifié de $\mathrm{B}_3 en B_5$.

$$\textbf{Parent} = \boxed{B_1}\boxed{B_2}\boxed{B_3}\boxed{B_4}\boxed{B_5} \longrightarrow \textbf{Fils} = \boxed{B_1}\boxed{B_2}\boxed{B_5}\boxed{B_4}\boxed{B_5}$$

Permuter deux gênes consécutifs : Les gênes de positions 3 et 4 sont échangés.

$$\textbf{Parent} = \boxed{B_1}\boxed{B_2}\boxed{B_3}\boxed{B_4}\boxed{B_5} \longrightarrow \textbf{Fils} = \boxed{B_1}\boxed{B_2}\boxed{B_4}\boxed{B_3}\boxed{B_5}$$

Permuter deux gênes quelconques : Les gênes de positions 1 et 4 sont échangés.

$$\textbf{Parent} = \boxed{B_1}\boxed{B_2}\boxed{B_3}\boxed{B_4}\boxed{B_5} \longrightarrow \textbf{Fils} = \boxed{B_4}\boxed{B_2}\boxed{B_3}\boxed{B_1}\boxed{B_5}$$

Inverser l'ordre d'une sous chaine du chromosome : La sous chaine B2 B3 B4 B5 est inversée.

$$\textbf{Parent} = \boxed{B_1}\boxed{B_2}\boxed{B_3}\boxed{B_4}\boxed{B_5} \longrightarrow \textbf{Fils} = \boxed{B_1}\boxed{B_5}\boxed{B_4}\boxed{B_3}\boxed{B_2}$$

Mutation en codage binaire

Dans un algorithme génétique, la mutation en codage binaire est la modification aléatoire occasionnelle (de probabilité faible) de la valeur d'un caractère de la chaine.

Mutation en codage réel

Pour le codage réel, les opérateurs de mutation les plus utilisés sont les suivants :
- L'opérateur d'inversion simple : consiste à choisir aléatoirement deux points de coupure et inverser les positions des bits situés au milieu.
- L'opérateur d'insertion : consiste à sélectionner au hasard un bit et une position dans le chromosome à muter, puis à insérer le bit en question dans la position choisie.
- L'opérateur d'échange réciproque : cet opérateur permet la sélection de deux bits et les inter changés.

4.4 Optimisation multiobjectif et algorithmes évolutionnaires

Dans cette section, on va présenter les algorithmes évolutionnaires les plus utilisés pour la résolution des problèmes d'optimisation multiobjectif.

4.4.1 Sélection multi-critères

Toutes les techniques de sélection présentées ci-dessus concernent le cas d'une fonction objectif à valeurs réelles. Cependant le problème posé consiste à sélectionner les individus mais en tenant compte de plusieurs critères au lieu d'un seul. On considère que la technique de base consistant à réunir tous les critères sous la forme d'une somme pondérée ne répond pas correctement à ce problème. Il faut donc adapter d'autres opérateurs de sélection.

Front de Pareto

Dans un problème multi-critère dans lequel on cherche à optimiser plusieurs objectifs contradictoires, on appellera front de Pareto du problème l'ensemble des points de l'espace de recherche tels qu'il n'existe aucun point qui est strictement meilleur qu'eux sur tous les critères simultanément. Il s'agit de l'ensemble des meilleurs compromis réalisables entre les objectifs contradictoires, et l'objectif de l'optimisation va être d'identifier cet ensemble de compromis optimaux entre les critères.

Sélections de Pareto

Lorsque l'on s'intèresse à l'optimisation multi-critère au sens de Pareto, il est possible de remplacer l'étape de sélection telle qu'elle a été décrite précedament par des sélections basées sur la notion de dominance au sens de Pareto. Cependant, la relation d'ordre définie par la dominance étant une relation d'ordre partiel, il faudra rajouter une procédure de choix secondaires entre individus non comparables au sens de Pareto.

Critères de Pareto

Le plus utilisé des critères de sélection au sens de Pareto est le rang de Pareto, défini itérativement de la manière suivante : les individus non-dominés de la population courante sont dits de rang 1. Ils sont retirés de la population, et la procédure est itérée pour obtenir les rang 2, 3, . . . Les individus de rang 1 (i.e. non dominés) vont approcher le front de Pareto du problème.

La force de domination est le nombre d'individus de la population courante qu'un individu donné domine : plus un individu domine d'autres individus, plus il est intéressant à conserver comme géniteur. De même, la profondeur au sens de Pareto d'un individu est le nombre d'autres individus de la population qui dominent un individu donné – un individu est d'autant plus intéressant à conserver qu'il est dominé par un petit nombre d'autres collègues.

Critères de diversité

Le critère précédent ne permettent pas d'ordonner toute la population – en fait, assez rapidement, les individus de la population courante qui vont approcher le front de Pareto du problème seront non comparables entre eux. Il faut donc ajouter un autre critère pour choisir entre eux. Les différents critères proposés favorisent la diversité. Le plus populaire aujourd'hui est la distance de peuplement ou la distance de Crowding.

Tournois au sens de Pareto

Pour sélectionner un géniteur a l'aide des outils définis ci-dessus, on utilise un tournoi en comparant tout d'abord les individus selon le critère de Pareto choisi, puis, en cas d'égalité, suivant le critère de diversité retenu.

4.4.2 Algorithmes évolutionnaires multiobjectif

Il est reconnu actuellement que les problèmes réels d'optimisation sont caractérisés par plus d'un objectif et ne peuvent donc pas être résolus par une technique d'optimisation à un seul objectif (mono-objectif).

Comme les algorithmes génétiques travaillent sur une population de points, il est naturel de les utiliser dans la résolution des problèmes d'optimisation miltiobjectif pour capturer une multitude de solutions simultanément. Ainsi, l'utilisation des méthodes évolutionnaires et les techniques d'optimisation parallèle telles que les AG, permettent de trouver un ensemble de solutions optimales dit de Pareto, c'est-à-dire, il n'y a pas de solution préférée à ces solutions sur l'ensemble des objectifs. Cependant, la propriété d'optimalité de Pareto suppose que les objectifs ont une importance égale. Réellement les utilisateurs, sont souvent capables de différencier les objectifs au travers de préférences par des règles de décision du processus de classement des solutions.

Parmi ces algorithmes évolutionnaires multiobjectif on parle du Non-Dominated Sorting Genetic Algorithm dans ces deux version NSGA II et NSGA III. Ces deux algorithmes sont détaillés dans la section qui va suivre.

NSGAII plus qu'un simple algorithme génétique

Selon Goldberg, (1989) et (Golchha and Qureshi, 2015), les tests de logiciels constituent la plus importante mesure de la qualité, qui consomme au moins 50des coûts de développement des logiciels. Le processus d'automatisation de la génération des données de test est un moyen de réduire le temps consommé par cette tâche. Les algorithmes génétiques sont utilisés dans ce but. Un certain nombre d'algorithmes évolutionnaires ont été suggérés. L'algorithme génétique de tri non dominé (NSGAII pour Non-Dominated Sorting Genetic Algorithm) est un algorithme utilisé pour résoudre des problèmes d'optimisation multicritères. Il a été proposé par Deb et al. (2002), en utilisant et améliorant le concept proposé par Goldberg (1989). NSGAII est l'un des algorithmes les plus utilisés pour résoudre les problèmes d'optimisation multicritères (Golchha and Qureshi, 2015).

Avant d'expliquer le fonctionnement de NSGAII, il est bon de préciser que NSGA a généralement été critiqué pour sa complexité de calcul, son manque d'élitisme et pour la difficulté à choisir la valeur de paramètre optimale pour le partage du paramètre σ_{share}(Seshadri, 2006). Deb et al. (2000, 2002) expliquent que NSGAII diffère de son prédécesseur par :

- l'utilisation d'une procédure de tri plus rapide, basée sur la non dominance ou Pareto optimale, avec une complexité de calcul de $O(mN^2)$, où m est le nombre d'objectifs et N la taille de la population.
- Une approche élitiste qui permet de préserver la diversité des populations, en sauvegardant les meilleures solutions trouvées lors des générations précédentes d'une part, et d'autre part un opérateur de comparaison basé sur un calcul de la distance de peuplement (ou Crowding) (Belaidi et al., 2009). La mise en œuvre de cet opérateur ne nécessite aucun réglage des paramètres.

Fonctionnement général de NSGAII

NSGAII est un algorithme élitiste qui n'utilise pas de dépôt (ou archive) externe pour stocker les meilleurs éléments. Pour gérer l'élitisme, NSGAII garantit que les meilleurs individus rencontrés soient conservés, à chaque nouvelle génération. Intéressons nous au fonctionnement de NSGAII, prenons une génération : où

deux populations P_t et Q_t de taille N cohabitant . La population P_t contient les meilleurs individus rencontrés jusqu'à la génération t, tandis que la population Q_t est constituée d'individus issus des phases précédentes du déroulement de NSGAII.

La première étape consiste à créer la population $R_t = P_t \cup Q_t$ et à appliquer une procédure de ranking pour identifier les différents fronts F_i de solutions non dominées. Par la suite, les

meilleurs individus se retrouvent dans le - ou les - tous premiers fronts).

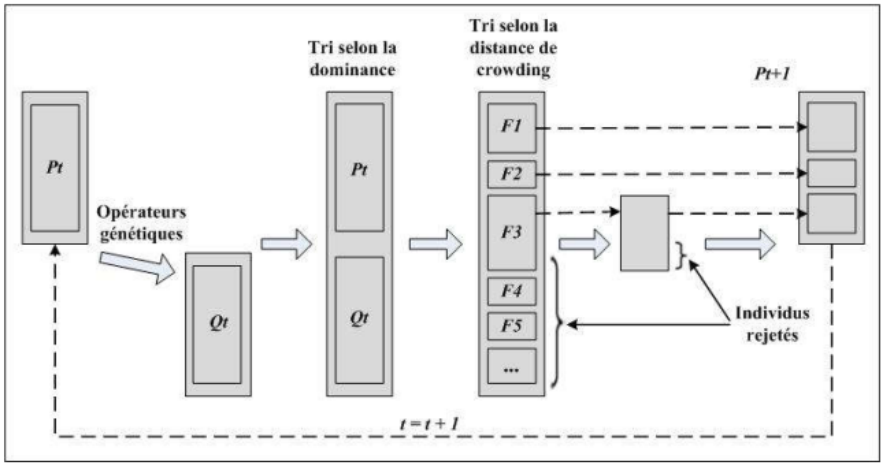


FIGURE 4.4 – Fonctionnement général de NSGAII

La deuxième phase consiste à construire une nouvelle population P_{t+1} contenant les N meilleurs individus de R_t . Pour y arriver, il est nécessaire d'inclure les meilleurs fronts F_i(complètement ; en commençant par l'indice 1, et ceci tant que le nombre d'individus présents dans P_{t+1} est inférieur à N. Il reste donc à ce stade $N-\mid P_{t+1} \mid$ individus à inclure dans P_{t+1}. Pour cela, une procédure de peuplement (ou crowding) est appliquée sur le premier front F_i non inclus. Donc, les $N-\mid P_{t+1} \mid$ meilleurs individus en utilisant la procédure de crowding sont insérés dans P_{t+1} .

La troisième phase se résume à emplir la population Q_{t+1}. Il suffit alors d'utiliser les opérateurs génétiques : Sélection, croisement et mutation (nous parlerons en détail des opérateurs utilisés par NSGAII, par la suite) sur les individus de P_{t+1} , puis d'insérer les descendants dans Q_{t+1}.

NSGAII est résumé dans l'algorithme suivant (Barichard, 2003) :

Distance de peuplement : crowding distance

Pour la sélection des individus à ajouter dans la population contenant les élites (P_{t+1}), Deb et al. (2000) ont proposé d'utiliser un paramètre particulier : La distance de crowding.

Pour calculer la distance de crowding d_i d'un point particulier i, il faut prendre en considération le périmètre de l'hypercube (ou n-cube) ayant comme sommets les points les plus proches de i sur chaque objectif. Une fois tous les d_i sont calculés, il ne reste plus qu'à les trier par ordre décroissant et à sélectionner les individus possédant la plus grande valeur de crowding.

Algorithme NSGAII
Initialiser les populations P_0 et Q_0 de taille N
Tant que *critère d'arrêt non rencontré* faire
- Créer $R_t = P_t \cup Q_t$
- Calculer les différents fronts F_i de la population R_t (ranking)
- Mettre $P_{t+1} = \emptyset$ et $i = 0$,
- **Tant que** $|P_{t+1}| + |F_i| < N$ faire
  $P_{t+1} = P_t \cup F_i$
  $i = i + 1$
  FinTantQue
- Inclure dans P_{t+1} les $(N - |P_{t+1}|)$ individus de F_i les mieux répartis selon la distance de crowding
- Sélectionner dans P_{t+1} et créer Q_{t+1} par application des opérateurs de croisement et mutation

FinTantQue

FIGURE 4.5 – Pseudo-code de NSGAII

L'algorithme génétique non dominé de tri NSGA III

NSGA III (Jian et Deb (2014) est un algorithme évolutionnaire basé sur son prédécesseur NSGA II. Le fonctionnement de NSGA III est le même que NSGA II. Cependant, NSGA III se distingue de NSGA II par la procédure de sélection des parents.

L'approche adoptée par NSGA III pour préserver le diversité est basée sur la répartition d'un ensemble de points de référence placé sur l'espace des objectifs. Les points de référence choisis peuvent être prédéfinis ou fournis en entrée de l'algorithme. En l'absence d'information sur les préférences, tout placement prédéfinis de points de référence peut être adopté. Dans (Jian et Deb (2014), les auteurs ont utilisé l'approche systématique de (Das and Dennis 1998)qui place les points sur un hyperplan normalisé défini par les vecteurs unitaires de dimension k . Soit p le nombre de divisions considéré pour chaque objectif, le nombre de points de références H générés pour un problème de k objectifs est donné par :

$$H = \begin{pmatrix} p+A-1 \\ p \end{pmatrix} \qquad (4.4)$$

En plus de privilégier les solutions non dominées, l'opérateur de sélection privilégie également les solutions associées à chacun de ces points de référence.

Les étapes de l'algorithme NSGA III sont :

étape 1 (*N*ormalisation de la fonction Objectif). on commence par la normalisation de la fonction objectif tout en utilisant la point idéal et le point extrême. On commence par détermination du

point idéal $z^{\min} = (z_1^{\min}, ..., z_A^{\min})$ qui est donné par la valeur minimale de la fonction objectif de chaque solution de la population S_t. ensuite on fait la soustraction du point $z^{\min}$.

$$f_a^{'}(x) = f_a(x) - z_a^{\min} \qquad (4.5)$$

Avec :

$a = 1, ...A$ dans notre cas $A = 2$, et $f_a(x)$ est la valeur de la fonction objectif de la solution x.

Le point extrême est identifié par la solution qui minimise la fonction ASF. On a :

$$\min ASF(x, w_a) = \max_{a=l}^{A} \frac{f_a^{'}(x)}{w_a}, \qquad (4.6)$$

avec :
$w = (w_1, \ldots, w_A)$: vecteur des poids w_i

pour trouver le a^{im} extrême point, on pose $w_a = 1$, tandis que les autres poids sont fixés à une très faible valeur 10^{-6}. On utilise A extrême points pour obtenir un hyperplan linéaire de dimension A. La fonction $f_a^{'}(x)$ devient

$$f_a^n(x) = \frac{f_a^{'}(x)}{c_a - z_a^{\min}}, \qquad (4.7)$$

où c_a est l'interception avec le a^{im} axe

étape 2 (*G*énérer des Points de référence) : des points de référence sont générés dans hyperplan normalisé par l'utilisation de l'approche systématique de Das et Dennis's [23]. pour A objectifs et p divisions on a :

$$H = \begin{pmatrix} p+A-1 \\ p \end{pmatrix} \qquad (4.8)$$

étape 3 (*c*alcul de la distance perpendiculaire) :

Après avoir normalisé la fonction objectif et générer les points de référence, il va falloir calculer la distance perpendiculaire entre la valeur objectif de chaque solution et la ligne de référence. Une solution de la population S_t est associée au point de référence correspondant à la distance minimale.

étape 4 (*O*peration de préservation) :

Soit ρ_h un paramètre qui donne le nombre de solutions dans S_t/F_v associées avec le h^{th} point de référence et on pose $J_{min} = \min \rho_h$. Le point de référence avec J_{min} est choisi.

- si $J_{\min} > 1$,alors un point de référence l^{th} est choisi aléatoirement .

- si $\rho_l \geq 1$ et le l^{th} point de référence est associé avec plus d'une solutions dans F_v, une solution dans F_v est sélectionnée aléatoirement et ajoutée à la population P_{t+1} et la valeur de ρ_l est décrémenté de 1. si $\rho_l \geq 1$ et aucune solution de F_v n'est associée à l^{th} point de référence, alors ce point n'est plus considéré dans la génération t.

-si $\rho_l = 0$ et le l^{im} point de référence point est associé avec plus d'une solution dans F_v, la solution avec une distance perpendiculaire minimale est ajoutée à la population P_{t+1} et ρ_l augmentera de 1. si $\rho_l = 0$ et aucune solution de F_v n'est associée à l^{th} point de référence, alors ce point n'est plus considéré dans la génération t

étape 5 (*O*pérations génétiques) :

Une fois la population P_{t+1} est obtenue, la population d'enfants Q_{t+1} est générée tout en utilisant les opérateurs génétiques de croissement et de mutation mentionnés dans l'algorithme NSGA II.

5

Métaheuristique Evolutionnaire Multiobjectif

Sommaire

5.1 Introduction

La résolution de différentes types de problèmes dans notre vie quotidienne a poussé la recherche pour proposer des méthodes de résolution fiables et à réaliser de grands efforts par les chercheurs pour améliorer leurs performances en termes de temps de calcul nécessaire et/ou de la qualité de la solution proposée. Dans ce chapitre, nous proposons une métaheuristique évolutionnaire multiobjectif pour la résolution du problème d'optimisation de portefeuille financier. Tous les résultats trouvés seront présentés sous forme de tableaux et courbes.

A la fin du chapitre, on présente les résultats trouvés par d'autres chercheurs ayant utilisé d'autres méthodes pour résoudre le même problème. Ces résultats seront comparés aux notre afin de déterminer la méthode la plus performante.

5.2 Sélection de portefeuille multiobjectif

Quel portefeuille est le meilleur ? Cette question est probablement aussi ancienne que la bourse elle-même. Cependant, lorsque Markowitz a publié son article sur la sélection des portefeuilles en 1952, il a donné les bases de la théorie moderne du portefeuille en tant que problème mathématique . Dans cette section on va évoquer le modèle de mean variance de Markowitz tout en rajoutant d'autres contraintes.

5.2.1 Description du problème

Supposons qu'il y ait n actifs financiers, dont les rendements sont donnés par les variables aléatoires R_1, ..., R_n.

Soit $Y = (y_1, y_2, y_3, ..., y_n)^T$, y_i est défini comme étant une variable binaire qui va prendre la valeur 1 si l'actif i est inclus dans le portefeuille en question et elle prends la valeur 0 dans le cas contraire.

On a :

$$y_i = \begin{cases} 1 \quad \text{si} \quad \text{i} \quad \text{est inclus dans le porteuille} \\ \\ 0 \quad \text{sinon} \end{cases} \tag{5.1}$$

avec :

$$\sum_{i=1}^{n} y_i = n \tag{5.2}$$

Rendement total : Le rendement total du portefeuille est donné par :

$$R_p = \sum_{i=1}^{n} R_i y_i \qquad (5.3)$$

Rendement espéré : Le rendement espéré du portefeuille est donné par [65] :

$$E(R_P) = E\left[\sum_{i=1}^{n} R_i y_i\right] = \sum_{i=1}^{n} E(R_i) y_i = \sum_{i=1}^{n} \mu_i y_i \qquad (5.4)$$

avec :

$E(R_i) = \mu_i$: Le rendement espéré pour l'actif numéro i.

Risque du portefeuille : Par définition le risque du portefeuille est donné selon Harry Markowitz (1959) [65] par la variance du rendement.

$$Var(R_p) = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} Cov(R_i, R_j) y_i y_j = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \rho_{ij} y_i y_j \qquad (5.6)$$

Où :

ρ_{ij} est la covariance entre le i^{me} et le j^{me} actif.

Mathématiquement, maximiser le rendement espéré du portefeuille et minimiser son risque (modélisé par la variance) simultanément tout en respectant un certain nombre de contraintes, conduit à un problème stochastique qui peut être formulé comme suit :

$$\begin{cases} \max \sum_{i=1}^{n} \mu_i y_i & & & & \text{maximiser le rendement} \\ \min \sum_{i=1}^{n} \sum_{j=1}^{n} \rho_{ij} y_i y_j & & & & \text{minimiser le risque} \\ \sum_{i=1}^{n} y_i & = & K & & \text{constrainte de cardinalité} \\ y_i y_j & = & 0 & \forall\, (i,j) \in F & \text{constrainte de pré-affectation} \\ y_i & \in & \{0,1\} & \forall\, i \in E & \end{cases}$$

avec : μ_i : rendement espéré de l'actif$i$.
ρ_{ij} : covariance entre l'actif i et l'actif j.
K : le nombre d'actifs à sélectionner.

Ce problème consiste à trouver un portefeuille efficient, c'est à dire déterminer la meilleure combinaison d'actifs qui y à la fois maximise le rendement espéré (modélisé par la moyenne

des rendements d'actifs) et minimise le risque (modélisé par la variance) tout en respectons les contraintes imposées.

La première contrainte du modèle représente la cardinalité (CC), cette contrainte fixe le nombre d'actifs à considérer et à sélectionner dans notre portefeuille. Cette contrainte est donnée comme suit :

$$\sum_{i=1}^{n} y_i = K \qquad (5.7)$$

avec :
K : c'est le nombre d'actifs à sélectionner.

Dans la deuxième contrainte, il s'agit modéliser les préférences subjectives de l'investisseur. Si par exemple l'investisseur ne souhaite pas s'investir simultanément dans une paire d'actifs (i, j), le portefeuille ne doit pas inclure les deux actifs i et j en même temps , juste un seul actif d'entre eux (soit i, soit j) est considéré dans la constitution du portefeuille. Ce type de d'exigence est modélisé par une contrainte de pré-affectation et elle est formulée de la manière suivante :

$$y_i y_j = 0, \forall (i, j) \in F \qquad (5.8)$$

F : l'ensemble qui contient toutes les paires d'actifs (i, j) tel que i et j ne peuvent pas être considérés simultanément dans le même portefeuille.

5.2.2 L'algorithme proposé : P-ACO

Pour résoudre le problème présenté ci-dessus, on propose une Métaheuristique basée sur les algorithmes de colonies de fourmis multiobjectif Pareto Ant Colony Optimization (P-ACO) décrite précédemment en détails. L'algorithme P-ACO pour résoudre notre problème de sélection de portefeuille est donné par :

Algorithme 2 Procedure P-ACO

1 : **étape 0 :** : Initialisation de P-ACO :
2 : - Créer Γ fourmis ;
3 : - Inisiliser le vecteur de phéromone τ_0

4 : **étape 1 :** Itération :
5 : **for** ant=1 to Γ **do**
6 : - Fixer la durée de vie K de chaque fourmi (nombre d'actifs à sélectionner)
7 : - Soit $y = (0,0,\ldots,0)$: créer un portefeuille vide pour chaque fourmi, $\sum_i y_i = 0$ and $|y| = n$;
8 : - $sum := 0$: créer une variable sum qui donne le nombre d'actifs à rajouter à y ;
9 : - Déterminer aléatoirement dans $[0,1]$, les poids p_l pour chaque objectif l
10 : **while** $sum \leq K$ **do**
11 : - Sélectionner l'actif i selon la distribution de probabilité et le rajouter à y
12 : - mise à jour locale de phéromone
13 : - $sum := sum + 1$
14 : **end while**
15 : **end for**

16 : **étape 2 :** test de faisabilité
17 : **si** le portefeuille y est faisable **alors**
18 : tester l'efficacité de y
18 : **fin si**
19 : **si** le portefeuille y est efficace **alors**
20 : enregister le portefeuille y et supprimer les portefeuilles dominés
21 : **fin si**

5.3 Optimisation du portefeuille sélectionné

Le premier volet de ce travail consiste à trouver la meilleure sélection d'actifs possible intervenant dans la constitution du portefeuille efficient et ce par l'application de l'approche de Pareto Ant Colony Optimisation. Dans deuxième volet, il va falloir passer à la détermination des proportions budgétaires optimales allouées au portefeuille efficient. Pour cela on introduit les l'algorithmes génétiques NSGA II et NSGA III.

5.3.1 Modèle mathématique proposé

Le modèle ci après nous permettra de déterminer les proportions budgétaires optimales allouées aux actifs qui constituent le portefeuille, de telle sorte à avoir un maximum de rendement et un risque minimum. D'un autre, il faudra respecter les contraintes imposées sur le budget, la

cardinalité et l'affectation des actifs.

La contrainte budgétaire consiste à répartir la totalité du budget initial sur l'ensemble des actifs sélectionnés. En d'autres termes, la somme des proportions (poids) investies dans les différents actifs est égale à 1. Cette contrainte est formulée de la façon suivante :

$$\sum_{i=1}^{n} x_i = 1 \qquad (5.9)$$

La contrainte de cardinalité (CC) fixe le nombre d'actifs à considérer dans notre portefeuille. Cette contrainte est donée comme suit :

$$\sum_{i=1}^{n} y_i = K \qquad (5.10)$$

avec :
K : c'est le nombre d'actifs à sélectionner.

Dans la contrainte de pré-affectation on modélise les préférences subjectives de l'investisseur. Si par exemple l'investisseur ne souhaite pas s'investir simultanément dans une paire d'actifs (i, j), le portefeuille ne doit pas inclure les deux actifs i et j en même temps , just un seul actif d'entre eux (soit i, soit j) est considéré dans la constitution du portefeuille. Ce type de d'exigence est modélisé par une contrainte de pré-affectation et elle est formulée de la manière suivante :

$$y_i y_j = 0, \forall (i, j) \in F \qquad (5.11)$$

F : l'ensemble qui contient toutes les paires d'actifs (i, j) tel que i et j ne peuvent pas être considérés

Le modèle mathématique pour résoudre le problème est le suivant :

$$\begin{cases} \max \sum_{i=1}^{n} \mu_i x_i & \text{maximise le rendement} \\ \min \sum_{i=1}^{n} \sum_{j=1}^{n} \rho_{ij} x_i x_j & \text{minimise le risque} \\ \sum_{i=1}^{n} x_i = 1 & \text{constrainte du budget} \\ 0 \leq x_i \leq 1 \quad \forall\, i \in E & \end{cases}$$

5.3.2 L'algorithme proposé : NSGA II et NSGA III

Pour résoudre le problème présenté ci-dessus, on propose d'appliquer séparément les deux algorithmes génétiques de tri NSGA II et NSGA III.

Pour implémenter l'algorithme NSGA II, on suit les étapes suivantes : [53]

- Considérer un portefeuille d'actifs efficient généré par P-ACO.
- Créer un vecteur de taille K qui contient le rendement de chaque actif du portefeuille .
- Créer une matrice de dimension $K \times K$ de variances-covariance.
- Créer N_{pop} vecteurs de taille K contenant chacun les proportions x_i choisis aléatoirement dans $[0,1]$ puis les normaliser en utilisant la formule suivante : $\acute{x_i} = \frac{x_i}{\sum_{k=1}^{K} x_k}$. Pour rendre l'implémentations plus simple, il est préférable de regrouper ces vecteurs dans une matrice de dimension $N_{pop} \times K$.
- Calculer le rendement esperé de chaque solution en utilisant la formule : $\sum_{i=1}^{K} \mu_i x_i$.
- Calculer le risque de chaque solution en utilisant la formule $\sum_{i=1}^{K} \sum_{j=1}^{K} x_i x_j \rho_{ij}$.
- Trier les solutions par le principe de dominance et sélectionner N_{pop} solutions, On obtient une matrice de N_{pop} lignes, chaque ligne de cette matrice représente la solution sélectionnée, cette matrice est notée par M.
- Créer N_{pop} nouvelles solutions en utilisant l'opérateur de croisement arithmétique

$$(x, \acute{x}) \longmapsto \alpha x + (1-\alpha)\acute{x}, \quad \alpha \sim U[0,1]$$

  avec $\alpha = 0.46$. on note la matrice trouvée par MC. puis on applique l'opérateur de mutation pour générer la population d'enfants. .
- Normaliser la matrice contenant la population d'enfants puis la faire combiner avec la population des parents pour obtenir une population de taille $2N_{pop}$.
- Calculer le rendement et le risque de toutes les solutions puis faire un tri à base de dominance. Les solutions non dominées sont sélectionnées pour former le premier front, par la suite ces solutions seront ignorées pour pouvoir sélectionner les solutions non dominées du deuxième front. Ce processus est répété jusqu'à ce que tous les individus de la population soient affectés à un front.
- Une nouvelle population parent est formée en ajoutant les fronts au complet tant que ce ceux-ci ne dépassent pas N_{pop}.
- une nouvelle population enfant de taille N_{pop} est créée par sélection, croisement. le critère de sélection est maintenant basé sur l'opérateur « Crowded-Comparison »
- Le processus se répète d'une génération à une autre jusqu'à satisfaction d'un critère d'arrêt.

Pour NSGA III, une fois arriver à la détermination de tous les fronts F_i, l'algorithme applique un processus de sélection basé sur les points de références pour pouvoir choisir les solutions qui reste pour compléter la population et obtenir la population de parents $P_{(t+1)}$.

Pour implémenter l'algorithme NSGA III, on suit les étapes suivantes :

- normalisation de la fonction objectif tout en utilisant le point idéal et le point extrême.
- Générer H Points de référence, pour A objectifs et p divisions on a :

$$H = \begin{pmatrix} p+A-1 \\ p \end{pmatrix} \qquad (5.12)$$

- Calculer la distance perpendiculaire entre la valeur objectif de chaque solution et la ligne de référence.

– Sélectionner les solutions intervenant dans la population P_{t+1} grace à une opération de préservation des solutions.
– Appliquer les operateurs génétiques sur la population P_{t+1} et générer la population d'enfants Q_{t+1}. Pour l'opérateur de croisement on garde le croisement arithmétique. Pour l'opérateur de mutation, dans un premier temps, on garde la même méthode utilisée dans l'algorithme NSGA II. Par la suite en remplace l'opérateur de mutation par une méthode basé sur la recherche locale dont l'algorithme est donné ci dessous.

.

Algorithme 3 Algorithme de la recherche locale

1 : **pour** chaque solution x dans MC **faire**
2 : calculer la valeur objectif de chaque solution x
3 : **répéter**
4 : soit ε dans [0,1]
5 : calculer $x^{'} = x + \varepsilon$
6 : **si** $x^{'}$ est dominée par x **alors**
7 : $x := x$
8 : répéter $4,5,6$
9 : **fin si**
10 : **si** x est dominée par $x^{'}$ **alors**
11 : $x := x^{'}$
12 : répéter $4,5,6$
13 : **fin si**
14 : **si** x et $x^{'}$ sont équivalentes **alors**
15 : **réserver** x et $x^{'}$
16 : $x := x^{'}$
17 : répéter $4,5,6$
18 : **fin si**
19 : **fin répéter**
20 : **fin**

5.3.3 L'effet de cardinalité sur le modèle

Dans cette partie de travail, on s'intéresse à l'effet de la cardinalité sur le modèle proposé. Pour confirmer cette hypothèse, on propose de faire une analyse de la variance à un facteur ANOVA.

Analyse de la variance

A partir d'un échantillon de n individus, on regroupe la population en des groupes clairement identifiables grâce aux modalités des variables qualitatives. Ces variables sont appelées facteurs. En effet, quand on parle de facteur c'est qu'il y a un phénomène à expliquer. Et quand on parle

de variance c'est qu'on veut en déceler la précision de cette explication. L'ANOVA (analysis of variance) répond bien à cette problématique et suppose donc une certaine intuition que l'on souhaite vérifiée. On confirme ou non cette intuition par des tests statistiques après avoir formaliser le modèle. S'il y a une différence significative en moyenne entre les groupes, cette différence sera imputée, toute chose étant égale par ailleurs, à la variable qui a servi à catégoriser les individus statistiques

Les conditions d'application

L'homoscédasticité : L'homoscédasticité désigne le caractère d'une liaison entre deux variables lorsque la variance de Y est la même pour toutes les valeurs de X et réciproquement. Cette condition doit être remplie pour que toutes les prévisions faites sur Y à partir de X (ou sur X à partir de Y) aient le même degré de précision. Il existe plusieurs tests possibles en fonction des situations expérimentales rencontrées : (Test de levene, test de Breuch-Pagan, test de Goldfled et Quandt,est de Bartlett,test de Brown- Frosythe).

La condition de normalité : On doit s'assurer que les variables sont distribués selon la loi normale. Si cela est le cas, les tests d'hypothèse classiques sont applicables.

Test de shapiro wilk : Le test de Shapiro-Wilk est basé sur la statistique W : En comparaison des autres tests, il est particulièrement puissant pour les petits effectifs $(n < 50)$. La statistique du test s'écrit :

$$W = \frac{\left[\sum_{i=1}^{[\frac{n}{2}]} a_i (x_{(n-i+1)} - x_{(i)})\right]^2}{\sum_i (x_{(i)} - \bar{X})^2} \qquad (5.13)$$

Avec :
$x_{(i)}$ correspond à la série des données triées ;
$[\frac{n}{2}]$ est la partie entière du rapport $\frac{n}{2}$;
les a_i sont des constantes générées à partir de la moyenne et de la matrice de covariance des quantiles d'un échantillon de taille n suivant la loi normale. ces constantes sont fournies dans des tables spécifiques.

La statistique W peut donc être interprétée comme le coéfficient de détermination (le carré du coéfficient de corrélation) entre la série des quantiles générées à partir de la loi normale et les quantile empirique obtenues à partir des données, plus la compatibilité avec la loi normale est crédible [72].

La région critique, rejet de la normalité s'écrit :$R.C.$: $W < W_{crit}$ et les valeurs seuils W_{crit} pour différents risque α et effectifs n sont lues dans la table de Shapiro-Wilk.

Si la P-value est inférieure à un niveau α choisi alors l'hypothèse nulle est rejetée c'est à dire improbable d'obtenir de telle données en supposant qu'elles soient normalement distribuées.
Si P-value est supérieure au niveau α choisi alors on ne doit pas rejeter l'hypothèse nulle.

Principe de l'analyse de la variance

La comparaison de différentes populations est un des problèmes les plus courants de la statistique. Le but principal de l'analyse de la variante (Anova) est de comparer les moyennes de plusieurs populations vérifiant certaines conditions à partir d'échantillons prélevés dans ces populations.

Considérons que lors d'une expérience, nous nous intéressions à l'étude sur n unités expérimentales, des variations d'une variable y (nombre de solutions générées par exemple) en fonction d'un facteur étudié composé de$l$ de modalités bien définies (cardinalité par exemple) ; les modalités du facteur étudié sont affectées de manière aléatoire aux unités expérimentales.

Une telle expérience peut être modélisée par l'équation suivante :

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij} \qquad (5.14)$$

Avec :$\varepsilon_{ij} \sim iidN(0,\sigma^2)$ et $\sum_{i=1}^{l} \alpha_i = 0$, $i = 1,...,l$, $j = 1,...,n_i$ $n = \sum_i n_i$

y_{ij} étant la valeur de la variable aléatoire y observée sur la j^{im} unité recevant le modalité i (cardinalité i) ; μ est la moyenne générale ; α_i , appelé l'effet de la modalité i, est l'écart entre la moyenne de la modalité i (cardinalité i) et la moyenne générale ; et ε_{ij} est l'erreur résiduelle.

Afin de procéder à la comparaison des moyennes des traitements nous allons confronter, à partir des données observées, l'hypothèse nulle H_O qui consiste à affirmer qu'il n'y pas d'effet dû aux traitements (c'est ,A dire que les traitements sont identiques) et l'hypothèse alternative H_1 qui revient à dire que les traitements ne sont pas identiques.

On peut montrer qu'on obtient l'expression suivante :

$$\sum_i^l \sum_j^{n_i} (y_{ij} - \bar{y}_{..})^2 = \sum_i^l \sum_j^{n_i} (y_{ij} - \bar{y}_{i.})^2 + \sum_i^l \sum_j^{n_i} (y_{i.} - \bar{y}_{..})^2 \qquad (5.15)$$

$$SCT = SCM + SCR \qquad (5.16)$$

Avec :

$y_{i.} = \sum_j^{n_i} y_{ij}$ et $\bar{y}_{i.} = \frac{1}{n_i} y_{i.}$.

$y_{..} = \sum_i^l y_{i.}$.

$\bar{y}_{..} = \frac{1}{n} y_{..} = \frac{1}{n} \sum_i^l y_{i.} = \frac{1}{n} \sum_i^l \sum_j^{n_i} (y_{ij})$.

Ceci montre que la variation centrée des observations est la somme de la dispersion due aus facteur (SCM) et d'une dispersion aléatoire (SCR). Ces sommes de carrés d'écart seront utilisés dans le test de H_0 contre H_1.

Pour tester l'hypothèse H_0 on utilise la statistique :

$F = \frac{SCM/l-1}{SCR/n_l}$ suit une loi de Fisher $F(l-1, n-l)$ sous H_0 .

Pour un risque α fixe, la zone d'acceptation de H_0 est : $ZA_{H_0,\alpha} = [0, F(l-1, n-l; 1-\alpha)]$

5.4 Partie pratique et Résultats obtenus

Dans cette section, nous présentons les résultats de calcul obtenus en effectuant des expériences sur un ensemble de données accessible au public, L'approche présentée a été réalisée à partir de cinq ensemble de données de référence provenant de la bibliothèque de Baseley'OR . Ces données fournissent les données d'entrée nécessaires pour divers actifs dans différents indices boursiers HANG SENG avec 31 actifs, SP100 avec 98 actifs, SP500 avec 476 actifs, NASDAQ avec plus de 2196 actifs. Toutes les fonctions et les procédures informatiques ont été implémentées en R avec un ordinateur équipé s'un system Intel(R) core(TM) i5, 8th gen, 08 Go RAM, Windows 8.

On rappelle que l'objectif de ce travail est de résoudre un problème d'optimisation de portefeuille sous contraintes de budget, de cardinalité et pré-affectation. Par exemple, pour contrainte de cardinalité, on fixe le nombre d'actifs à sélectionner à $K = 10$ pour l'indice de Hang Seng et on choisi trois paires d'actifs dans l'ensemble F, $F = (16,17),(17,18),(16,18)$, On rappelle que l'ensemble F est l'ensemble qui contient toutes les paires d'actifs (i, j) tel que i et j ne peuvent pas être considérés simultanément dans le même portefeuille (contrainte de pré-affectation). Pour Nasdaq, on fixe $K = 10, 100, 300$ et on considère six paires d'actifs dans l'ensemble F (Table 5.1).

5.4.1 Résultats issus de P-ACO

Le bon fonctionnement de l'algorithme P-ACO nécessite certains paramètres pour obtenir une bonne qualité de solution et maintenir une évaluation contrôlable du processus d'optimisation (Table 5.2). L'application de l'approche P-ACO sur nos données donne les résultats suivants : [53]

TABLE 5.1 – Cardinalité et pré-affectation de chaque indice.

	Hang Seng	**SP 100**	**SP 500**	**Nasdaq**
Cardinalité (K)	10	50, 80	10, 100, 300	10, 300
Pré-affectation (F)	(16,17) (17,18) (16,18)	(15,33) (63,71) (73,77)	(38,73) (94,131) (150,175) (177,183)	(29,34) (37,40) (45,47) (64,65) (1413,1418) (1738,1758)

TABLE 5.2 – Les paramètres de P-ACO.

	Hang Seng	**SP 100**	**SP 500**	**Nasdaq**
Γ	10	30	50	5
α	1	1	1	1
ρ	0.7	0.7	0.7	0.7
τ_0	0.9	0.9	0.9	0.9

TABLE 5.3 – Portefeuille effecient généré par P-ACO pour Hang Seng.

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}
port 1	0	0	0	1	1	0	0	1	1	0	0	1	0	0	0	0
port 4	0	0	0	0	1	0	0	1	1	0	0	1	0	0	0	1
port 7	1	0	0	0	1	0	0	1	1	0	0	1	0	0	0	0

	p_{17}	p_{18}	p_{19}	p_{20}	p_{21}	p_{22}	p_{23}	p_{24}	p_{25}	p_{26}	p_{27}	p_{28}	p_{29}	p_{30}	p_{31}
port 1	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0
port 4	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0
port 7	1	0	1	1	0	0	0	0	0	1	0	0	1	0	0

	Return	**Risk**
port 1	0.058008	54.094946
port 4	0.053634	50.604180
port 7	0.050428	49.152482

Nous pouvons voir sur la (Table 5.3) que l'algorithme P-ACO utilise pour l'indice Hang Seng 10 fourmis, chaque fourmi est censée construire son propre portefeuille. Par la suite, tous les portefeuilles générés par les fourmis ont été triés selon le principe de pareto afin de garder

TABLE 5.4 – Risque et rendement du portefeuille effecient généré par P-ACO.

SP 100 with 98 assets		
K	**Return**	**Risk**
50	0.213265	512.678372
	0.112512	422.242630
	0.190705	503.768738
	0.116.808	423.820148
	0.176744	497.690060
	0.171635	495.680658
	0.131665	429.933680
	0.135661	433.616562
	0.149273	433.901566
	0.157965	441.118896
	0.161059	443.883738
	0.164927	450.406884
	0.168713	456.368526
80	0.278026	1207.943510
	0.256027	1159.959632
	0.234959	1115.002572
	0.262656	1165.628104
	0.249885	1131.073054
	0.271474	1184.533588
	0.255227	1159.089108
	0.234178	1110.485858
	0.248012	1121.461844
	0.224832	1108.411672
	0.266096	1178.966284
	0.255227	1138.903924
	0.214503	1097.030830

SP 500 with 476 assets		
K	**Return**	**Risk**
10	0.1354794	0.1376716
	0.09525867	0.10177683
	0.08501251	0.08373711
	0.06799201	0.03822819
	0.05084089	0.03679313
	0.04417696	0.03518582
	0.04118156	0.03308424
	0.02510106	0.02298574
	0.02734874	0.02596494
	0.02871795	0.0266791
100	0.7540985	5.2578543
	0.4622423	3.2788443
	0.3448385	2.8524634
	0.2410349	2.3486477
	0.6114916	4.1390230
	0.4323154	3.2468762
	0.3193307	2.6280283
	0.547265	3.500980
	0.4052092	3.1003410
	0.2950339	2.4596917
	0.5023697	3.3834249
	0.3770722	2.8849828
300	1.561179	32.089310
	1.174359	28.197236
	0.9588885	23.8943005
	1.415065	31.665333
	1.150750	25.919244
	1.389743	28.851918
	1.211427	28.791523

Nasdaq with 2196 assets		
K	**Return**	**Risk**
10	0.2080628	0.214315
	0.1503612	0.17065663
	0.1428687	0.1422794
	0.11326854	0.09698731
	0.09275083	0.08668862
	0.08419325	0.0847406
300	3.263283	55.474549
	2.021897	44.972060
	1.4616685	42.463714
	1.065910	32.841021
	0.7308886	29.8021161
	0.4063562	27.2410551
	2.270448	46.702699
	1.618908	43.829508

Hang Seng with 31 assets		
K	**Return**	**Risk**
10	0.058008	54.094946
	0.053634	50.604180
	0.050428	49.152482

uniquement les meilleurs portefeuilles et supprimer tous les portefeuilles dominés. A la fin de ce processus de tri, on remarque que processus garde uniquement 03 solutions non dominées c'est à dire 03 portefeuilles efficients composés chacun de 10 actifs. Sur la même table, on remarque aussi que la fourmi numéro 1 a construit le portefeuille avec le plus grand rendement et le plus grand risque. On verra par la suite que cette solution nous permettra de déterminer les solutions pour les proportions budgétaires allouées aux 10 actifs intervenant dans la constitution du portefeuille efficient.

Afin de générer plus de 03 portefeuilles on a pensé à modifier les valeurs des paramètres de

l'algorithme. Par exemple, augmenter le nombre de fourmis utilisées à 30 puis à 50 et enfin à 100. Les résultats trouvés après ces modifications été les même que ceux d'avant.

Table 5.4 donne les solutions optimales (risque et le rendement) générées par P-ACO pour chaque indice et relativement à chaque valeur de K.

Nous pouvons voir sur la figure que nous avons une population raisonnablement diverse et ce pour chaque indice. On remarque aussi que plus la valeur de K augmente plus la valeure du couple (rendement,risque) augmente. En effet, Pour l'indice SP 500, on trouve que la plus grande valeur du couple (rendement,risque) est (1.561179, 32.089310), elle corresponde à la plus grande valeur de K qui est 300. Pour Nasdaq aussi on trouve $(3.263283, 55.474549)$ pour $K = 300$.

Si l'investisseur favorise le rendement il n'a qu'a s'investir dans des grand portefeuilles de taille importante. En revanche, les investisseurs qui préfèrent pas prendre le risque doivent plutôt s'engager dans des petits portefeuilles car le risque sera plus petit dans cette situation.

5.4.2 Résultats issus de NSGA II et NSGA III

Dans cette sous section, on présente les résultats obtenus par l'implémentation des algorithmes de tri NSGA II et NSGA III. [53]

Le but dans cette partie de travail est la détermination les proportions budgétaires optimales allouées aux actifs qui constituent chaque portefeuille effecient, généré dans la première partie, par l'approche P-ACO. L'implémentation des algorithmes NSGA II et NSGA III nous permettra de déterminer ce capital budgétaire de telle sorte à avoir simultanément un maximum de rendement et un minimum de risque. Tout en respectant les contraintes imposées sur le budget, la cardinalité et l'affectation des actifs.

Les deux algorithmes ont été exécutés plusieurs fois, il été nécessaire de fixer certains paramètres pour obtenir une bonne qualité de solution.

- Taille de la population : $taille - pop = 50$.
- Critère d'arrêt : $iteration - max = 50, 100, 200$.
- Probabilité de croisement $P_c \in \{0.2, 0.9\}$.
- probabilité de mutation : $P_m = 0.7$.

L'implémentation des deux algorithmes à donné les résultats suivants :
Les figures ci dessus montrent les frontières efficientes trouvées pour, Hang-Seng, SP100, SP500 et Nasdaq selon les différentes valeurs de K.

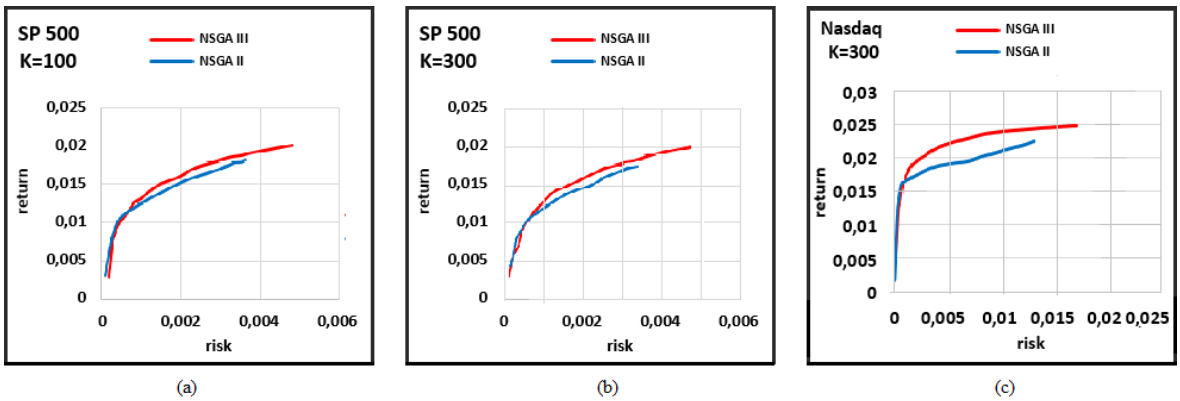
Le choix de la meilleure combinaison d'actifs du portefeuille, la probabilité de croissement et la probabilité de mutation, fixées à l'avance, contribuent énormément important sur la qualité de la solution. En effet, si on fixe les probabilités de croissement et de mutation respectivement

TABLE 5.5 - Nombre de solutions non dominée et temps d'exécution de NSGA II et NSGA III.

	Hang Seng 31 actifs		SP100 98 actifs					
	NSGA II	NSGA III	NSGA II			NSGA III		
K	10	K	10	50	80	10	50	80
ITE_MAX	50	ITE_MAX	100	100	100	100	100	100
TE (sec)	19.092	TE (sec)	29.457	35.1050	38.2561	39.5631	40.772	46.5941
NS	27	NS	24	19	20	32	27	28

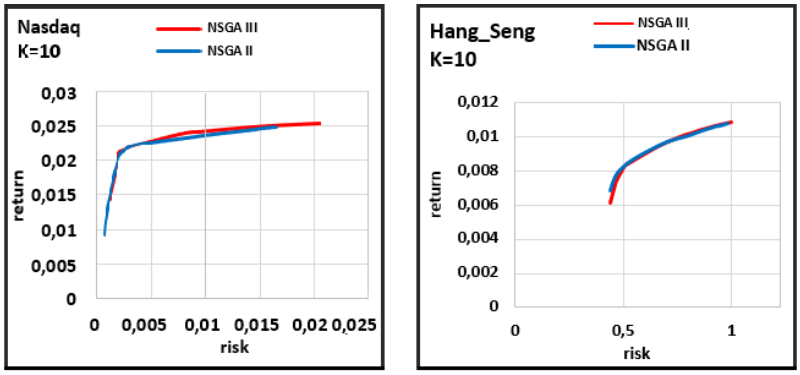
	SP500 476 actifs							
	NSGA II				NSGA III			
K	10	80	100	300	10	80	100	300
ITE_MAX	200	200	200	200	200	200	200	200
TE (min)	50.149 s	1.303083	1.42	2.207769	57.838 s	1.395067	1.5307	2.7037
NS	16	21	19	21	39	38	33	34

	Nasdaq 2196 actifs							
	NSGA II				NSGA III			
K	10	80	100	300	10	80	100	300
ITE_MAX	200	200	200	200	200	200	200	200
TE (min)	8.12795	8.256967	8.135533	9.437117	8.12795	8.256967	8.135535	9.437117
NS	19	20	19	19	36	38	30	39



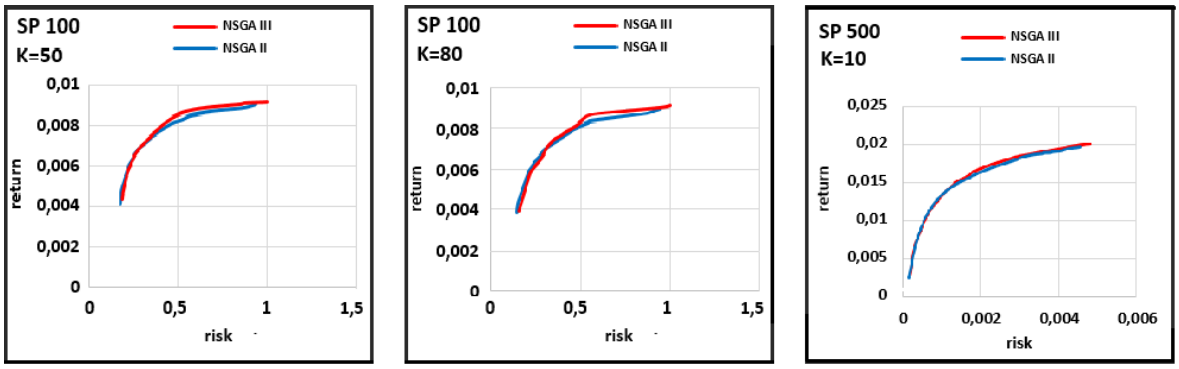
(a) (b) (c)

FIGURE 5.1 – Solutions non dominées de SP500 et Nasdaq pour $K \in \{100, 300\}$.



(a) (b)

FIGURE 5.2 – Solutions non dominées de Nasdaq et Hang_Seng pour K=10.



(a) (b) (c)

FIGURE 5.3 – Solutions non dominées de SP 100 pour $K \in \{50, 80\}$ et SP 500 pour K=10.

à $P_c = 0.90$, $P_m = 0.7$, ceci nous permettra de tracer la frontière efficiente entre $]0, 0.001]$ pour le risque et entre $]0.022]$ pour le rendement. Par contre, une probabilité de croissement plus élevée $P_c = 0.90$ nous a permis de tracer la frontière efficiente entre $[0.002, 0.016]$ pour le risque et entre $[0.015, 0.025]$ pour le rendement.

Pour SP 100 et Hang Seng , on a diminué la probabilité de croissement jusqu'a 0.2 et garder la même probabilité de mutation pour trouver de bonne qualité de solutions.

On conclue donc que le choix des valeurs fixées pour les probabilités de croissement et de mutation ont un rôle important dans l'exploration de l'espace de recherche et la diversité des solutions.

D'un autre côté, les figures et la table 5.5 montrent que NSGA III est plus performant que NSGA II en termes de nombre de solutions générées et en qualité aussi. En effet, pour SP500

et NASDAQ (les portefeuilles de taille importante). En effet, Avec NSGA III on obtient un rendement maximale pour SP500 (K=300) égale à 0,020114381 alors que le rendement maximal obtenu par NSGA II est de 0,01816846. Pour Nasdaq (K=300) NSGA III trouve un rendement égale à 0,02488537 et NSGA II trouve 0,02249477.

Résultat obtenus par la Recherche locale

Si on considère l'algorithme de la recherche locale comme un opérateur génétique de mutation dans l'implémentation de NSGA III, on trouve les résultats suivants :

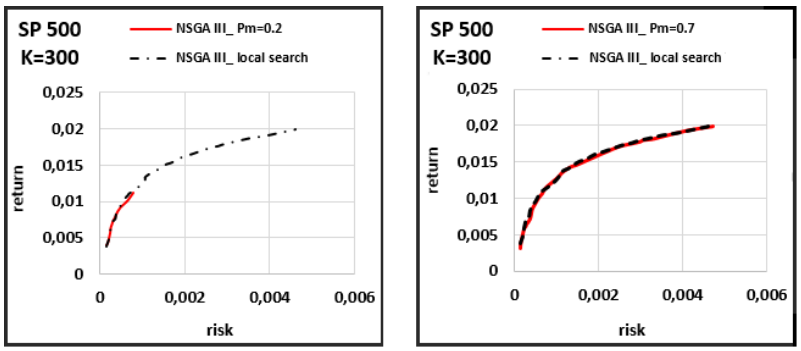


FIGURE 5.4 – Frontière des solutions non dominées obtenues par NSGA III pour SP500

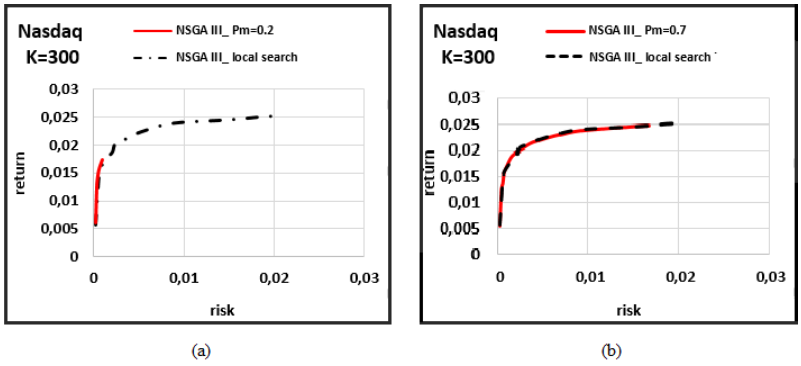


FIGURE 5.5 – Frontière des solutions non dominées obtenues par NSGA III pour Nasdaq

Les figures ci-dessus montrent les solutions trouvées en basant sur la recherche locale, utilisée en tant que opérateur génétique de mutation dans NSGA III, sont meilleurs par rapports aux solutions trouvées précédemment par NSGA III avec le mécanisme de mutation classique et une probabilité de mutation 0.2. Donc on peut conclure que la recherche locale représente une bonne alternative dans le fonctionnement des algorithmes génétiques.

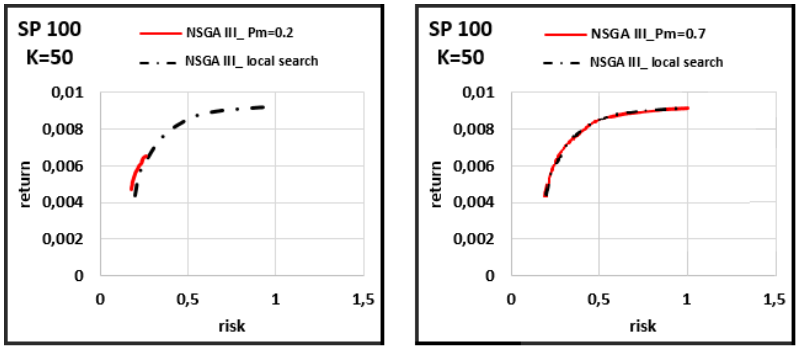


FIGURE 5.6 – Frontière des solutions non dominées obtenues par NSGA III pour SP 100

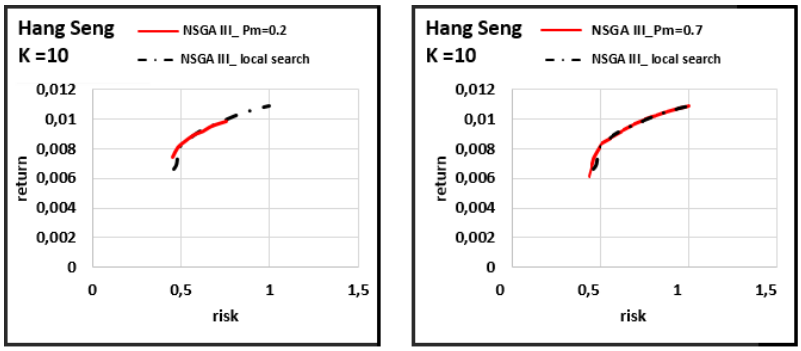


FIGURE 5.7 – Frontière des solutions non dominées obtenues par NSGA III pour Hang Seng

Par contre avec $P_m = 0.7$, les solutions obtenues par les deux mécanismes sont équivalentes. Encore une fois la probabilité de mutation influence sur la diversité des solutions.

L'effet de cardinalité sur l'algorithme NSGA III

Dans cette partie, on s'intéresse à linfluence de la contrainte de cardinalité sur le nombre de solutions générées par NSGA III. pour cela on une analyse de la variance à un facteur.

Le facteur étudié est la cardinalité (nombre d'actifs sélectionné), les différentes modalités de ce facteur sont : Dix, quatre-vingt, cent, trois cent et la variable dépendante est le nombre de solutions dominées trouvées par NSGA III (NS).

Nos hypothèses H_0 est H_1 sont définies comme suit :

H_0 : il ya aucune différence entre les solutions de NSGA III trouvées en fonction des diffé-

rentes modalités de K. H_1 : il ya au moins une solution différente due au facteur .

Résultat de l'ANOVA

Test de Normalité : test de Shapiro Wilk

Dans le test de Shapiro Wilk l'hypothèse H_0 de normalité est vérifiée quand la valeur de statistique W calculée est supérieur au W critique de la table ; On trouve, à l'aide du langage R, ce résultat :
shapiro.test(X)
Shapiro-Wilk normality test :
data : X
$W = 0.9332$, p-value = 0.4444

On remarque que P value est supérieure au niveau α ; ce qui confirme la validité de l'hypothèse H_0. Alors on accepte H_0, donc les données suivent une distribution normale.

Tester l'égalité des variances : test de Fisher les hypothèses du test sont :
comme on a quatre échantillons donc on refait six fois le test :

H_0 : Le rapport entre les variances est égal à 1. H_1 : Le rapport entre les variances est différent de 1.

Résultat du test

Echantillon 1 et 2
F (Valeur observée) : 0,250
F (Valeur critique) : 39,165
DDL1 : 3
DDL2 : 2
p-value (bilatérale) : 0,0,285
alpha : 0,05

Echantillon 1 et 3
F (Valeur observée) : 1,852
F (Valeur critique) : 864,163
DDL1 : 3
DDL2 : 1
p-value (bilatérale) : 0,969
alpha : 0,05

Echantillon 1 et 4
F (Valeur observée) : 0,667

F (Valeur critique) : 864,163
DDL1 : 3
DDL2 : 1
p-value (bilatérale) : 0,616
alpha : 0,05

Echantillon 2 et 3
F (Valeur observée) : 7,407
F (Valeur critique) : 799,500
DDL1 : 2
DDL2 : 1
p-value (bilatérale) : 0,503
alpha 0,05

Echantillon 2 et 4
F (Valeur observée) : 2,667
F (Valeur critique) : 799,500
DDL1 : 2
DDL2 : 1
p-value (bilatérale) : 0,795
alpha : 0,05

Echantillon 3 et 4
F (Valeur observée) : 0,360
F (Valeur critique) : 647,789
DDL1 : 1
DDL2 : 1
p-value (bilatérale) : 0,688
alpha : 0,05

Etant donné que toutes les la p-value calculées sont supérieures au niveau de signification seuil $\alpha = 0,05$, on ne peut pas rejeter l'hypothèse nulle H_0.

ANOVA

Source	DDL	Somme des carrés	Moyenne des carrés	F	Pr > F
Modèle	3,000	29,515	9,838	0,634	0,616
Erreur	7,000	108,667	15,524		
Total corrigé	10,000	138,182			

D'après la valeur de la P value qui est égale à 0,616 et qui est supérieure au seuil $\alpha = 0.05$, on déduit qu'il n ya pas de différence significative entres les solutions générées par NSGA III selon les différentes valeurs de K (cardinalité).

5.5 Comparaison avec une méthode Exacte

Dans cette section, on compare les résultats obtenus par NSGA III avec ceux d'une autre méthode [12]utilisée dans le même contexte et appliquée sur les mêmes données .

Bezoui et al., (2018) ont proposé une variante de "Epsilon constraint" pour la résolution du problème d'optimisation de portefeuille sous contraintes. La méthode a été implémentée sur les mêmes données que nous avons utilisés dans notre travail. [12]

Nous avons copié leurs frontières efficientes présentées pour les différents indices et pour les mêmes valeurs de k et voila ce qu'on a obtenu.

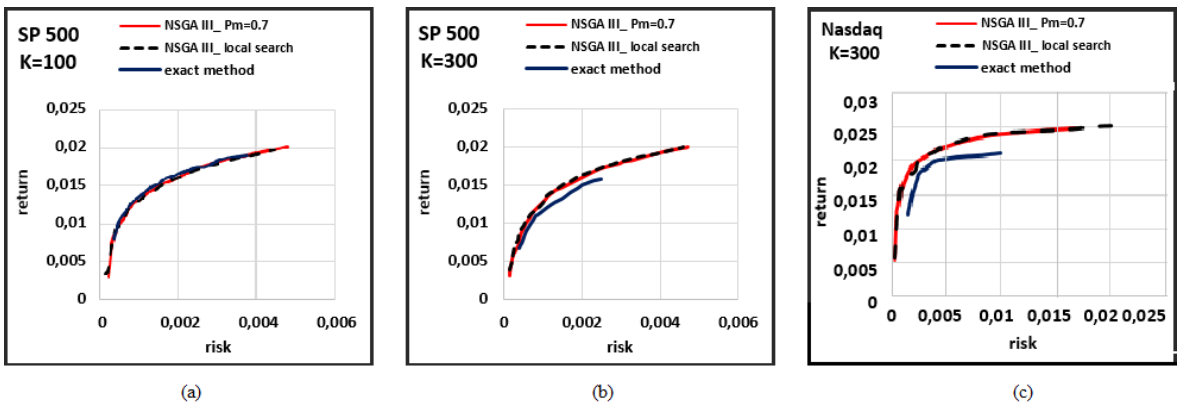


FIGURE 5.8 – Frontières efficientes obtenues par les approches pour SP500 (K= 100 and K=300) et pour Nasdaq (K=300)

La figure 5.8 montre les frontières efficientes trouvées pour SP500 et NASDAQ par les deux approches, selon les deux valeurs $K = 100$ et $K = 300$. D'après les courbes, on voit que notre approche évolutionnaire est meilleure que la méthode exacte utilisée par les auteurs. En effet, le rendement maximum trouvé pour SP 500 (K=300) par la méthode itérative (exacte) égal à 0.016 est beaucoup plus petit que le rendement trouvé par notre métaheuristique évolutionnaire (0.02). La même chose pour Nasdaq, on trouve un rendement de 0.025,ce qui n'est pas le cas pour la méthode itérative.

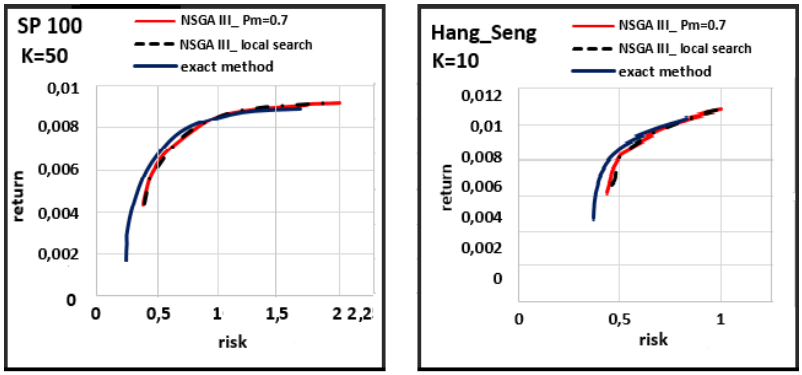


FIGURE 5.9 – Frontières efficientes obtenues par les approches pour SP100 (K=50)et HANG SENG (K=10)

Par contre sur la figure 5.9, on remarque que les deux approches donnent pratiquement les mêmes résultats. On peut donc dire que l'utilisation des algorithmes de colonies de fourmis et algorithmes génétiques (NSGA II, NSGA III) donnent des résultats meilleurs que ceux de la méthode itérative notamment quand il s'agit d'une masse importante de données.

Conclusion générale

Dans cette thèse, le problème d'optimisation de portefeuille est considéré sous sa forme bi-objectif (rendement et risque) et traité sous trois contraintes : contrainte de budget, contrainte de cardinalité et contrainte de pré-affectation.

Pour résoudre ce problème, nous avons proposé une approche métaheuristique évolutionnaire multi-objectifs basée sur la métaheuristique Pareto Ant Colony Optimization (P-ACO) et les algorithmes de tri génétique non-dominés (NSGA II et NSGA III). P-ACO est utilisée pour sélectionner les meilleurs actifs composant le portefeuille efficient. Ensuite, NSGA II et NSGA III sont utilisés séparément pour trouver les poids proportionnels du budget alloué au portefeuille sélectionné. Les résultats obtenus par ces deux algorithmes ont été comparés pour désigner l'algorithme le plus performant. Une autre comparaison est réalisée entre les résultats obtenus et ceux d'une méthode exacte utilisée pour le même problème. Les expériences numériques effectuées sur un ensemble d'instances issues de la littérature ont révélé que la combinaison de la métaheuristique d'optimisation par colonies de fourmis et de l'algorithme génétique NSGA III proposée donne le plus souvent de bien meilleurs résultats que la combinaison de la métaheuristique d'optimisation par colonies de fourmis et de l'algorithme génétique NSGA II.

Perspectives

Les recherches menées dans le cadre de la présente thèse peuvent être étendues dans d'autres directions. En effet, ce travail ouvre des perspectives théoriques appliquées dans la suite des études. La principale perspective théorique est la généralisation de la méthode proposée pour résoudre des problèmes d'optimisation sous contraintes et avec plus de trois objectifs rencontrés en Finance .

Bibliographie

[1] A. Hitaj and G. Zambruno, *Portfolio optimization using modified herfindahl constraint*, International Series in Operations Research Management Science, Springer, Berlin, Germany, pp. 211–239, 2018.

[2] Affleck-Graves J, MacDonald B : *Non-normalities and tests of asset pricing theories*, Journal of Finance, 1989.

[3] Alaya I, Solnon C, Ghédira K : *Optimisation par colonies de fourmis pour le problème du sac à dos multidimensionnel*, thèse de doctorat. 2005.

[4] Anagnostopoulos K P, Mamanis G : *A portfolio optimization model with three objectives and discrete variable*, Computers Operations Research 378, 1285–1297, 2010.

[5] Andersson J : *A Survey for Multiobjective Optimization in engineering Design*, Rapport technique, LiTH-IKP-R-1097, 2000.

[6] B. Zhang, T. Jiang, X. Zhou, and J. Duan, *A new method of portfolio optimization : mean-Co Va R model*, Statistics and Decision, vol. 35, no. 11, pp. 67–70, 2019.

[7] Back D, Fogel B, Michalewicz Z : *Handbook of Evolutionary Computation*, Oxford University Press, 1997.

[8] B. A. *Mercangöz, Portfolio optimization*, International Series in Operations Research and Management Science, vol. 306, pp. 15–27, 2021.

[9] Barbulescu L, J P Watson and Whitley D : *Dynamic Representations and Escaping Local Optima Improving Genetic Algorithms and Local Search*, 2000.

[10] Beasley J E, Or-library : *Distributing test problems by electronic mail*. J Oper Res Soc 41, 1069–1072. 1990

[11] Ben Abdelaziz F, Krichen S : *A tabu search heuristic for multiple objective knapsack problems*. Ructor Research Report RR, 1997.

[12] Bezoui M, Moulaï M, Bounceur A, Euler R : *An iterative method for solving a bi-objective constrained portfolio optimization problem*. Computational Optimization and Applications, 2018.

[13] Bullnheimer B, Hartl R F, Strauss C : *An Improved Ant system Algorithm for the Vehicule Routing Problem*, Annals of Operations Research, 1999.

[14] Cesarone F, Scozzari A, Tardella F : *Linear vs. quadratic portfolio selection models with hard real-world constraints*, Computational Management Science 12, 1–26, 2014.

[15] Cesarone F, URL : http ://w3.uniroma1.it/tardella/datasets.html. 2017

[16] Coello A c, and Becera r : *Evolutionary multiobjective optimization using acultural algorithm.*, IEEE Swarm Intelligence Sumposium Proceedings, 2003.

[17] Coello, A C, Lamont G B, Veldhuizen D A V : *Evolutionary Algorithms for Solving Multi-Objective Problems.*, Springer, Second edition, 2007.

[18] Collette Y, and Siarry P. *Multiobjective Optimization*. Springer, 2003.

[19] Colorni A, Dorigo M, Maniezzo V. *An investigation of some proprieties of an ant algorithm*. In Manner and Manderick, 1992.

[20] Cornuejols G : *Optimization Methods in Finance*, Carnegie Mellon University, Pittsburgh, PA 15213 USA.2006.

[21] Costanzo A, Luong V, MARILL G : *Optimisation par colonies de fourmis*, thèse de doctorat. 19 mai 2006.

[22] Czyzzak P, Jaszkiewicz A : *Pareto simulated annealing a metaheuristic technique for multiple-objective combinatorial optimisation*. Journal of Multi-Criteria Decision Analysis, 1998.

[23] Das I, Dennis J.E. Normal-bounday intersection : *A new method for generating pareto optimal points in multicriteria,optimization problems*, 1998

[24] Deb K : *Multi-Objective Optimization using Evolutionary Algorithms*, 2001.

[25] Deb K, Jain H : *An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach*, IEEE Trans. Evol. Comput 18, 2014

[26] Deb K, Pratap A, Agarwal S, Meyarivan T : *A Fast and Elitist Multiobjective Genetic Algorithm NSGA-II*, 182-197, 2002.

[27] Dejong W, Spears M : *A formel analysis of the role of multi-point crossover in genetic algorithms*, Annals of Mathematics and Artificial Intelligence Journal, 1992.

[28] Doerner K F, Gutjahr W J, Hartl R F, Strauss C, Stummer C : *Pareto ant colony optimization with ilp preprocessing in multiobjective project portfolio selection*. European Journal of Operational Research 171, 830–841. 2006.

[29] Doerner K, Gutjahr W J, Hartl R, Strauss C, Stummer C : *Pareto Ant Colony Optimization : A Metaheuristic Approach to Multiobjective Portfolio Selection*. Kluwer Acadeemic Publishers, 2002.

[30] Dorigo M : *Optimization, Learning and Natural Algorithms*, Ph.D. thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1992.

[31] Dorigo M , DICARO G : *The Ant Colony Optimization Meta-Heuristic*, Mc-Graw Hill, UK, 1999.

[32] Dorigo M, Stutzle T : *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.

[33] Dorigo M , Gambardella L M : *Ant Colony System : A Cooperative Learning Approach to the Traveling Salesman Problem* , IEEE Transactions on Evolutionary Computation, 1(1), 53-66, 1997.

[34] Dréo J, Petrowski , Taillard E, Siarry P : *Métaheuristiques pour l'optimisation difficile*, 2003.

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

[36] Fama E F : *The behaviour of stock market prices*, Journal of Buisness, vol.38,1965.

[37] Fonseca C M, Fleming P J : *Genetic algorithms for multi-objective optimization : formulation, discussion and generalization*. The Fifth International Conference on Genetic Algorithms, 416-423, 1993.

[38] Gambardella L M, Taillard E D, Agazzi G, *Macs-vrptw : A multiple ant colony system for vehicle routing problems with time windows, in : New Ideas in Optimization*, pp. 63–76. 1999.

[39] Gandibleux X, Mezdaoui N, Fréville A : *A multiobjective tabu search procedure to solve combinatorial optimization problems* , Advances in Multiple Objective and Goal Programming, Lecture Notes in Economics and Mathematical Systems, 291-300. 1997

[40] Gherboudj A : *Méthodes de résolution de problèmes difficiles académiques*, Université de Constantine,2012/2013.

[41] Ghosh A, Das M K : *Non-dominated rank based sorting genetic algorithms*. Fundamenta Informaticae 83. 2008.

[42] Goldberg D : *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA : AddisonWesley, 1989.

[43] Goldberg D : *Real-coded Genetic Algorithms, Virtual Alphabets, and Blocking*, University of Illinois at Urbana–Champaign, Technical Report, No. 90001, September 1990.

[44] Goldberg D : *Algorithmes Génétiques exploration, optimisation et apprentissage automatique*, Edition Addition-Wesley, 1994.

[45] Goubault E. *Cours sur les Algorithmes Évolutionnaires et Problèmes Inverses*, , École Polytechnique ParisTech, France, 2016.

[46] Gravel M, Price W L, Gagne C : *Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic*. European Journal of Operational Research 143, 218–229. 2002.

[47] H. A. Shalan and H. M. Abd El-Salam, *Multi-objective portfolio optimization using Pareto dominance and differential evolution*, Applied Soft Computing journal, January 2014.

[48] Haimes Y , Freedman H T : *A multiobjective optimization in water resources systems*, Elsevier Scientific, 1975.

[49] Hansen M P *Metaheuristics for multiple objective combinatorial optimization*, Ph.D. Thesis, Technical University of Denmark, Lyngby, 1998.

[50] Holland J, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[51] Horn J, Nafpliotis N, Goldberg D.E : *A niched pareto genetic algorithm for multiobjective optimization.* IEEE Service (ed), First IEEE Conference on Evolutionary Computation, vol. 1. Piscataway : IEEE World Congress on Computational Computation,82-87, 1994.

[52] Ibanez M L, *Multi-objective Ant Colony Optimization*. Ph.D. thesis. Technische Universität Darmstadt, Germany, and Universidat de Granadan Spain. 2004.

[53] Ikhlef M, Aïder M, *Multiobjective evolutionary metaheuristic approach to the constrained portfolio optimization problem*, Pesquisa Operacional,43 : e266962 P.1-23,doi : 10.1590/0101-7438.2023.043.00266962. 2023

[54] Jaszkiewicz, A . *On the Performance of Multiple-Objective Genetic Local Search on the* $0/1$ *Knapsack Problem-A Comparative Experiment* . IEEE Transactions on Evolutionary Computation, 6(4),2002, 402-412.

[55] Jorion P. : *Value at risk : the new Benchmark for controlling market Risk*, JMcGraw-Hill, New-york, 1997.

[56] Keeney R L, Raaiffa H : *Decision with Multiple Objective : Preference and value Tradoff.* Cambridge University Press, 1993.

[57] Kirkpatrick S, Gelatt J C D, Vecchi M P : *Optimization by Simulated Annealing*, 1983.

[58] Kolomvos G : *Résolution de grands problèmes stochastiques multi-étapes :Application à un problème de dimensionnement de capacités et de gestion de flux et de stocks*, école centrale des arts et manufactures, école centrale Paris, 2007.

[59] Koza J R : *Genetic Programming, on the Programming of Computers by means of Natural Selection*, MIT Press, 1992.

[60] Kumar D, Mishra K K : *Portfolio optimization using novel co-variance guided artificial bee colony algorithm.* Swarm and Evolutionary Computation 33, 119–130. 2017.

[61] Lalanne C, Pallier C : *Introduction aux Statistiques et l'utilisation du logiciel R*.

[62] Lawrence J, Alvin J. Michael J : *Artificial Intelligence through Simulated Evolution*, Wiley, 1966.

[63] Liu Q, Liu X, Wu J, Li Y : *An improved nsga-iii algorithm using genetic k-means clustering algorithm*. IEEE Access 7, 185239–185249. doi :10.1109/ACCESS.2019.2960531. 2019

[64] Lutton E : *Algorithmes génétiques et Fractales*. Habilitation 0 diriger des recherches, Université Paris XI Orsay, France 1999.

[65] Markowitz H. : *Portfolio selection, efficent diversification of investments*. J. Wiley, 1959.

[66] Miettienen K M : *Proper pareto Optimality in Nonconvex Problems- characterization with Tangent and Normal Cones*, Technical report, 1998.

[67] Moeini M : *La programmation DC et DCA pour l'optimisation de portefeuille*. Université Paul Verlaine-Metz, 2008.

[68] Morgan J P : *RiskMetrics* , Technical document 4th edition, 1996.

[69] N. O. Alsharif and A. O. Alghamdi, *A multi-objective approach to portfolio optimization using genetic algorithms* , Journal of Computational Science, January 2018.

[70] Osyczka A : *Multicritaria optimization for engineering design.*, Design optimization. Academic press, Cambrige, 193-227. 1985

[71] Pirlot M : *Métaheuristiques pour l'optimisation combinatoire*, Hermés Science Publications, Paris, 2002.

[72] Ricco Rakotomalala R : *Tests de normalité. techniques empiriques et tests statistiques*. Université Lumière Lyon. 2008.

[73] Sharpe W F : *A linear programming approximation for the general portfolio analysis problem*. Journal of Financial and Quantitative Analysis, 1263-1275, 1971.

[74] Schaffer J D : *Multiple objective optimization with vector evaluated genetic algorithms*, ICGA International Conference on Genetic Algorithms,93-100, 1985.

[75] Serafini P :*Simulated annealing for multiple objective optimization problems.*, Tenth Int. Conf. on Multiple Crit Decision Making, 87-96, 1992.

[76] Solnon C : *Ants can Solve Constraint Satisfaction Problems.*, IEEE Transactions on Evolutionary Computation, 6(4), 347-357. 2002.

[77] S. S. Islam and S. Das, *Multi-objective particle swarm optimization for portfolio selection*, Journal of the Expert Systems with Applications, September 2015.

[78] Syswerda G : *Uniform Crossover in Genetic Algorithms*, In Proceedings of the 4th International Conference on Genetic Algorithms, pp. 279-286, John Wiley and Sons, New York, 2nd edition, 1995.

[79] Takahashi R H C, Deb K, Wanner E F, Greco v : *Evolutionary Multi-Criterion Optimization*, 6th International Conference, EMO 2011,Ouro Preto, Brazil, April 2011.

[80] Ulungu E L, Teghem J, Frtemps P, Tuyttens D : *MOSA method : A tool for solving multi-objective combinatorial optimization problems*, Laboratory of Mathematic and Operational Research, Faculté polytechnique de Mons, 1998.

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

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

[83] Wright A, *Genetic Algorithms for Real Parameter Optimization*, First Workshop on the Foundations of Genetic Algorithms and Classifier Systems, Morgan Kaufmann Publishers, San Mateo, CA, pp. 205-218, 1991.

[84] Y. Zhang, Y. Li, and Q. Wang, *A multi-objective optimization approach for portfolio selection using NSGA-II* , Journal of the Expert Systems with Applications, March 2012.