REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE FERHAT ABBAS SETIF 1
FACULTE DES SCIENCES
DEPARTEMENT DE MATHEMATIQUES

Thèse présentée pour l'obtention du diplôme de

Doctorat LMD 3ème Cycle

Filière : Mathématiques

Option : Optimisation et Contrôle

Par

Manal HEDID

Thème

Résolution d'un problème de transport flou à quatre indices

Soutenue le : 03/12/2022

devant le jury composé de :

M. Mohamed ACHACHE	Prof.	Université Ferhat Abbas Sétif 1	Président
M. Rachid ZITOUNI	Prof.	Université Ferhat Abbas Sétif 1	Rapporteur
M. Abdelaziz AMROUNE	Prof.	Université Mohamed Boudiaf M'Sila	Examinateur
M. Abdelkrim MERZOUGUI	Prof.	Université Mohamed Boudiaf M'Sila	Examinateur
Mme. Hassina GRAR	MCA	Université Ferhat Abbas Sétif 1	Examinateur
M. Djamel BENTERKI	Prof.	Université Ferhat Abbas Sétif 1	Invité

Année universitaire : 2022/2023

Remerciements

En tout premier lieu, je remercie *ALLAH* le tout puissant de m'avoir donné la force pour survivre, ainsi que l'audace pour dépasser toutes les difficultés.

Je tiens à exprimer mes remerciements et ma gratitude à Monsieur Mr. Rachid ZITOUNI, Professeur à l'université Ferhat Abbas-Sétif 1, pour avoir assumé la responsabilité de m'encadrer, m'orienter et de me conseiller tout au long de la réalisation de ce travail, aussi pour sa disponibilité et son aide précieuse, accepte Monsieur mes remerciements les plus sincères.

Je tiens à remercier tout particulièrement, Monsieur Mohamed ACHACHE Professeur à l'université Ferhat Abbas-Sétif 1, pour l'honneur qu'il m'a fait de faire partie de ce jury et d'en être le président.

Mes vifs remerciements s'adressent aussi à l'ensemble des membres du jury : Monsieur Abdelaziz AMROUNE Professeur à l'université Mohamed Boudiaf-M'Sila, Monsieur Abdelkrim MERZOUGUI Professeur à l'université Mohamed Boudiaf-M'Sila, Madame Hassina GRAR MCA à l'université Ferhat Abbas-Sétif 1, d'avoir accepté de juger ce travail et d'en être les examinateurs. Aussi pour l'intérêt qu'ils ont porté à cette thèse ainsi que pour leurs précieuses remarques.

Aussi je voudrais remercier vivement Monsieur Adnan YASSINE, Professeur à l'université du Havre Normandie, pour son accueil chaleureux au sien de laboratoire de mathématiques appliquées du Havre, ainsi que pour ses nombreuses remarques.

Un très grand merci à toute ma famille : mes chers parents, mon beau-père que son âme repose en paix, ma belle-mère, mes sœurs et mes frères qui m'ont gratifié de leur amour et m'ont fourni les motivations qui ont permis l'aboutissement de mon travail doctoral. Je leur adresse toute ma gratitude du fond du cœur.

Mes derniers remerciements, et non les moindres, vont à mon mari Sami LABDAI, tu as toujours été là quand j'avais besoin de toi. Merci pour ton soutien, ta compréhension et ton sacrifice durant l'élaboration de cette thèse.

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

TABLE DES MATIÈRES

Liste des tableaux

Liste des figures

Introduction

L'optimisation est une branche importante des mathématiques consiste à trouver la plus petite valeur (problème de minimisation) ou la plus grande valeur (problème de maximisation) d'une fonction sur un ensemble. Elle intervient dans de nombreux domaines :

- En recherche opérationnelle (problème de transport, économie, gestion de stocks, ..., etc.),
- En analyse numérique (approximation/résolution de systèmes linéaires, non linéaires,..., etc.),
- En automatique (modélisation de systèmes, filtrage,..., etc.),
- En ingénierie (dimensionnement de structures, conception optimale de systèmes, réseaux, ordinateurs,..., etc.)

Les méthodes de résolution des problèmes d'optimisation peuvent être classées en deux catégories bien distinctes. D'une part, les méthodes exactes, cherchant à trouver de manière certaine la solution optimale en considérant l'ensemble des solutions possibles. D'autre part, les méthodes approchées, qui se contentent de rechercher une solution de bonne qualité. Généralement, les méthodes exactes prennent un temps considérable pour résoudre les problèmes d'optimisation notamment pour ceux de grandes tailles, ce qui limite leur utilisation en pratique. Les métaheuristiques sont introduites pour la résolution approchée des problèmes de grandes tailles. Elles forment un ensemble de méthodes utilisées en recherche opérationnelle et en intelligence artificielle pour résoudre des problèmes d'optimisation réputés difficiles.

Parmi les problèmes d'optimisation, on trouve le problème de transport (PT). C'est en 1941 que Frank L. Hitchcock [18] a formulé pour la première fois ce problème, qui consiste à minimiser le coût de transport total d'un plan d'expédition. Un tel problème est un modèle important de programmation linéaire qui se pose dans plusieurs contextes et a mérité une attention particulière. Bien qu'il peut être résolu en utilisant la méthode du simplexe [11], plusieurs chercheurs s'intéressent beaucoup à trouver d'autres méthodes plus adaptées, par le faite que l'application du simplexe pour résoudre un PT est très difficile. En effet, dans l'algorithme du simplexe on doit chaque fois utiliser des tableaux (ou matrices) de taille proportionnelle à la taille du problème.

L'algorithme le plus populaire pour résoudre le problème de transport se fait en deux phases. L'idée de cet algorithme c'est la recherche dans la première phase d'une solution de base réalisable initiale et dans la deuxième de la solution optimale. Les méthodes les plus connues dans la littérature permettant d'initialiser le problème de transport, c.-à-d. trouver une solution de base réalisable initiale, sont : la méthode du coin nord-ouest, la méthode du coût minimum, la méthode d'approximation de Russell et la méthode d'approximation de Vogel. Pour obtenir une solution optimale, une deuxième phase est nécessaire ; dans celle-là on apporte des améliorations successives à une solution de base réalisable jusqu'à ce qu'il n'y ait plus de réduction du coût de transport. Les méthodes largement utilisées pour trouver la solution optimale sont la méthode de Stepping-Stone et la méthode des distributions modifiées (MODI).

Le problème de transport a connu plusieurs extensions telles que : le problème de transport à trois indices [40], le problème de transport à quatre indices [29,43–46], le problème de transport multi-objectifs [2], le problème de transport flou [16,17,21], . . . , etc.

Ces dernières années, l'attention s'est concentrée sur la résolution du problème de transport dans un environnement flou où les données du problème sont complètement ou partiellement présentées sous forme de nombres flous. Il s'agit d'une application plus cohérente du problème au monde réel, en raison de l'outil de mesure imprécis, des données perdues et manquantes, et même des erreurs de calcul. Le problème de transport flou (PTF) est un problème dont les données sont incertaines et représentées par des approximations floues (nombres flous).

Notre objectif dans cette thèse est la résolution d'un problème de transport flou à quatre indices par deux types de méthodes : exactes et approchées. Elle est divisée en trois chapitres, et est organisée de la manière suivante. Dans le premier chapitre, nous présenterons quelques notions de base de la programmation linéaire, du problème de transport à quatre indices et des préliminaires sur les mathématiques flous (les ensembles et les nombres flous, ainsi que les opérations arithmétiques agissant sur les nombres flous et la fonction de classement).

Le second chapitre présente la première partie de notre contribution, il s'agit de l'adaptation des méthodes exactes classiques (coût minimum, approximation de Russell et approximation de Vogel) dans le contexte flou pour déterminer une solution de base réalisable initiale au problème de transport flou à quatre indices (PTF4) en trois types. Ensuite, on donne une adaptation de la phase 2 de l'algorithme AL_{PT4C} [44] dans le contexte flou afin de déterminer la solution optimale. On termine ce chapitre par une implémentation numérique en donnant un exemple pour chaque type, ainsi qu'une étude comparative entre ces différentes méthodes.

Le troisième chapitre est consacré à la deuxième partie de notre contribution, il s'agit de la résolution du PTF4 par les méthodes approchées ; On commence par la description et les fondations nécessaires à la compréhension des algorithmes génétiques (GA) ainsi que l'algorithme d'optimisation par essaim de particules, en anglais "Particle Swarm Optimization (PSO)". Ensuite, on présente deux approches pour résoudre le PTF4 : la première est une adaptation de GA et la deuxième est une hybridation des deux algorithmes GA et PSO. En termine ce chapitre par

une étude numérique comparative afin de tester les performances de nos approches vis-à-vis à la méthode exacte. Les résultats obtenus montrent la supériorité de la méthode hybride surtout pour les problèmes de grande taille.

Liste des symboles et abréviations

Symboles	Description
PL	Programme linéaire
PT	Problème de transport
PT4	Problème de transport à quatre indices
PTF	Problème de transport flou
PTF4	Problème de transport flou à quatre indices
AL_{PT4C}	L'agorithme itroduit par R. Zitouni pour résoudre le problème de transport à quatre indices avec capacités.
AL_{PT4}	La version sans capacités de l'algorithme AL_{PT4C}
M	Nombre de contraintes
N	Nombre de variables
FLC4	L'algorithme adapté de celui du coût minimum
FRAM4	L'algorithme adapté de celui d'approximation de Russell
FVAM4	L'algorithme adapté de celui d'approximation de Vogel
$\tilde{A}$	Nombre flou
$F(\mathbb{R})$	L'ensemble des nombres flous
μ	La fonction d'appartenance
$\Re(\cdot)$	La fonction de classement d'un nombre flou
$\oplus$	L'opération d'addition floue
$\ominus$	L'opération de soustraction floue
$\otimes$	L'opération de multiplication floue
AG	Algorithme génétique
PSO	Particle Swarm Optimization
$Pbest$	La meilleure solution obtenue par une particule
$Gbest$	La meilleure solution obtenue par toutes les particules

CHAPITRE 1

GÉNÉRALITÉS ET PRÉLIMINAIRES

Dans ce chapitre, nous rappelons les notions de base liées à notre étude qui seront utiles par la suite. Il est divisé en trois grandes parties ; dans la première, nous allons introduire certaines notions et certains résultats de la programmation linéaire. Dans la deuxième partie, nous allons présenter le problème de transport à quatre indices et la dernière partie est consacrée aux préliminaires des ensembles et des nombres flous, ainsi que les opérations arithmétiques agissant sur les nombres flous et la fonction de classement.

1.1 Programmation linéaire

La programmation linéaire constitue la pierre angulaire de toute la recherche opérationnelle. Elle peut se définir comme une technique mathématique permettant de résoudre des problèmes de gestion et particulièrement ceux où le gestionnaire doit déterminer, face à différentes possibilités, l'utilisation optimale des ressources de l'entreprise pour atteindre un objectif spécifique comme la maximisation des profits ou la minimisation des coûts.

Dans la plupart des cas, les problèmes de l'entreprise peuvent être traités par la programmation linéaire impliquant un certain nombre de ressources par exemple, main-d'œuvre, matières premières, ..., qui sont disponibles en quantité limitée et qu'on veut répartir d'une façon optimale entre un certain nombre de processus de fabrication.

Un programme linéaire (PL) s'exprime de façon standard sous la forme

$$\min \left[c^T x : \ x \geq 0, \ Ax = b\right] \qquad (P_0)$$

avec $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ et $A \in \mathbb{R}^{m \times n}$.

1.1.1 Définitions

Étant donné un programme linéaire de la forme P_0.

— on appelle "solution" tout vecteur x tel que $Ax = b$ et "solution réalisable" ("admissible") tout vecteur x tel que $Ax = b$ et $x \geq 0$.

— Supposons que $rang(A) = m$, on appelle "base B", toute sous-matrice carrée d'ordre m régulière (inversible) extraite de A.
— On associe à toute base B, les décompositions $A = [B \mid N]$ et
$x = \begin{pmatrix} x_B \\ x_N \end{pmatrix}$ Si l'on fixe le vecteur x_N à zéro, alors le vecteur
$x = \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix}$ vérifiant $Ax = b$ est dit "solution de base".
— Une solution de base est dite "réalisable" si $x_B \geq 0$.
— Les composantes du vecteur x_B sont appelées "variables de base" et les composantes du vecteur x_N sont appelées "variables hors-base".
— Une solution de base est dite "dégénérée" si certaines de ses variables de base sont nulles.
— Une solution de base est dite "optimale" si elle rend la fonction objectif (économique) $c^T x$ optimale.

1.1.2 Dualité

La dualité est un concept fondamental en programmation linéaire, elle conduit à un résultat de grande portée théorique et pratique.

Définition 1.1.1. *Le dual du programme linéaire P_0 est le programme linéaire suivant*

$$\max \left[b^T y : \ A^T y \leq c, \ y \in \mathbb{R}^m \right] \qquad (D_0)$$

Le programme P_0 est dit primal.

Propriétés fondamentales de la dualité

Étant donné deux programmes linéaires duaux sous les formes P_0 et D_0, on a :

Lemme 1.1.2. *Si x est une solution réalisable de P_0 et y est une solution réalisable de D_0, alors*

$$c^T x \geq b^T y.$$

Corollaire 1.1. *Soient $\bar{x}$ une solution réalisable de P_0 et $\bar{y}$ une solution réalisable de D_0 telles que*

$$c^T \bar{x} = b^T \bar{y},$$

alors $\bar{x}$ est une solution optimale de P_0 et $\bar{y}$ est une solution optimale de D_0.

Théorème 1.1.3. (**Théorème de la dualité**) *: Une condition nécessaire et suffisante pour qu'une solution réalisable x^* (ou y^*) de l'un des programmes duaux, soit optimale est qu'il existe une solution*

réalisable y^* (ou x^*) de l'autre programme telle que

$$c^T x^* = b^T y^*,$$

cette solution y^* (ou x^*) du deuxième programme est aussi optimale.

Théorème 1.1.4. (**Théorème faible des écarts complémentaires**) *: Si x et y sont respectivement deux solutions réalisables du primal et du dual, alors elles sont optimales si et seulement si :*

$$(Ax - b)^T y = 0 \text{ et } (c - A^T y)^T x = 0.$$

1.1.3 Résolution d'un programme linéaire

Méthode graphique : C'est l'une des premières méthodes utilisées pour déterminer la solution optimale d'un programme linéaire à deux ou trois variables au plus, dont la solution optimale atteinte en un sommet du polyèdre convexe. Mais d'un point de vue pratique, cette méthode a beaucoup de limites par rapport aux variables à traiter.

Méthode de Simplexe : Elle a été introduite par George Dantzig à la fin des années 40. Elle comporte deux phases ; d'abord trouver une solution réalisable (sinon détecter l'impossibilité), puis passer d'un sommet à un sommet voisin en améliorant la valeur de l'objectif jusqu'à l'optimum.

Méthodes de points intérieurs : Ces méthodes ont été introduites à la fin des années 50 afin de résoudre des programmes mathématiques non linéaires à cause de la dominance quasi-totale de la méthode du simplexe à cette époque. Après l'apparition de l'algorithme de Karmarkar en 1984 [19] pour la programmation linéaire, les méthodes de points intérieurs ont connu une véritable révolution.

Il y a trois classes fondamentales de méthodes de points intérieurs : les méthodes affines, les méthodes de réduction du potentiel et les méthodes de trajectoire centrale.

1.1.4 Applications

Les domaines d'application de ces problèmes sont très nombreux aussi bien dans la nature des problèmes abordés (planification et contrôle de la production, distribution dans des réseaux) que dans les secteurs d'industrie : industrie manufacturière, énergie (pétrole, gaz, électricité, nucléaire), transports (aériens, routiers et ferroviaires), télécommunications, industrie forestière, finance, ...

Parmi les modèles d'application de la programmation linéaire les plus connus dans la littérature, on trouve le problème de transport. Depuis son apparition, ce problème a connu différents

modèles et extensions. Dans la section suivante, on présente une de ses extensions.

1.2 Problème de transport à quatre indices

Plusieurs chercheurs se sont intéressés à ce problème ; il s'agit d'une modélisation pratique appropriée pour résoudre des problèmes de la vie réelle avec des offres, des demandes, ainsi que des différents moyens de transport et des différentes qualités de produits.

Parmi les premiers travaux concernant le problème de transport à n-indices on trouve les recherches de K. B. Haley en 1963 [15]. L'auteur a défini le problème pour $n = 3$. Il a aussi proposé un algorithme pour le résoudre, cet algorithme est une extension de la méthode de MODI. M. Queyranne et F. C. R. Spieksma (1997) [31] ont traité un cas particulier du problème de transport axial à indices multiples où les coûts sont décomposables, ils ont présenté deux algorithmes pour le résoudre.

Dans leurs travaux, R. Zitouni et al. [43–46] ont apporté des réponses intéressantes concernant le problème de transport à quatre indices avec capacité. Ils ont établi les conditions de réalisabilité et le critère d'optimalité de tel problème ce qui leur permis de construire l'algorithme AL_{PT4C} pour sa résolution. H. Bulut et S.A. Bulut (2003) [7] ont étudié la construction et la formulation algébrique d'un problème de transport à quatre indices. Pham et al. (2013) [29] visait à trouver une solution sur le coût de transport minimum pour le problème de transport à quatre indices. Ils ont également développé une méthode exacte pour résoudre ce problème. Wang et al. (2018) [40] ont mis en uvre une méthode de décomposition basée sur la modification séquentielle du critère d'optimalité pour traiter le problème de transport à trois indices. Ils ont étudié les solutions de ce problème dans le cas d'une fonction objectif linéaire ou quadratique. D. S. Sarode et R. Tuli (2019) [32] ont implémenté la méthode de distribution modifiée pour obtenir une solution optimale pour le problème de transport à quatre indices. K. Tanwar, S.K. Chauhan (2019) [39] ont traité le problème de transport bi-objectifs à indice multiples comme une extension d'un problème de transport à indices multiples avec objectif unique. Leur méthode a été présentée pour minimiser le temps et le coût simultanément. Skitsko et al. [38] ont développé une approche pour appliquer l'algorithme génétique pour résoudre le problème transport à quatre indices.

Une méthode de détecter l'ensemble complet non dominé (solutions efficaces) des problèmes de transport multi-objectifs à quatre indices a été présentée par A. E. M. Abd Elazeem et al. (2021) [2]. L'approche proposée met en uvre une méthode de somme pondérée pour convertir ce problème en un problème de transport mono-objectif à quatre indices, qui peut ensuite être décomposé en un ensemble de sous-problèmes de transport à deux indices.

1.2.1 Position du problème

Le problème de transport à quatre indices est formulé mathématiquement comme suit :

$$\begin{aligned}
&\text{Minimiser } Z = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} c_{ijkl}x_{ijkl} \\
&\text{sous les contraintes :} \\
&\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \alpha_i \ \text{ pour tout } \ i = 1,\dots,m, \\
&\sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \beta_j \ \text{ pour tout } \ j = 1,\dots,n, \\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q} x_{ijkl} = \gamma_k \ \text{pour tout } \ k = 1,\dots,p, \\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} x_{ijkl} = \delta_l \ \text{pour tout } \ l = 1,\dots,q, \\
&x_{ijkl} \geq 0, \ \text{pour tout } i = 1:m, j = 1:,n, k = 1:p, l = 1:q.
\end{aligned} \tag{1.1}$$

Avec $\alpha_i > 0, \beta_j > 0, \gamma_k > 0, \delta_l > 0$, et $c_{ijkl} \geq 0$, $\forall(i,j,k,l)$.

Ce problème peut être formulé sous forme d'un programme linéaire comme suit :

$$\begin{cases}
\min Z = c^T x \\
\text{sous les contraintes :} \\
Ax = b \\
x \geq 0
\end{cases} \tag{1.2}$$

Où

- $x = (x_{1111},\dots,x_{mnpq})^T \in \mathbb{R}^N$,
- $c = (c_{1111},\dots,c_{mnpq})^T \in \mathbb{R}^N$,
- $b = (\alpha_1,\dots,\alpha_m,\beta_1,\dots,\beta_n,\gamma_1,\dots,\gamma_p,\delta_1,\dots,\delta_q)^T \in \mathbb{R}^M$,
- A est une $M \times N$ matrice,
- $M = m+n+p+q$ et $N = mnpq$.

Tableau de transport

Est un outil de présentation des données d'un problème de transport à quatre indices. C'est un tableau de M lignes, N colonnes, deux lignes marginales et une colonne marginale. Les cases de ces N colonnes de la première et la deuxième ligne marginale sont réservées aux valeurs des quantités, c_{ijkl}, et x_{ijkl} respectivement. Les cases de la colonne marginale sont réservées aux valeurs des quantités $\alpha_i, \beta_j, \gamma_k$, et δ_l respectivement. Finalement, l'entrée de la case du tableau située à la ligne correspondant à α_{i0} et à la colonne P_{ijkl} vont 1 si $i = i_0$ et 0 ailleurs. De même pour β_{j0}, γ_{k0}, et δ_{l0}.

c_{1111}	c_{1211}	$\cdots$	c_{1npq}	c_{2111}	c_{2211}	$\cdots$	c_{2npq}	$\cdots$	c_{m111}	c_{m211}	$\cdots$	c_{mnpq}	
x_{1111}	x_{1211}	$\cdots$	x_{1npq}	x_{2111}	x_{2211}	$\cdots$	x_{2npq}	$\cdots$	x_{m111}	x_{m211}	$\cdots$	x_{mnpq}	
1	1	$\cdots$	1	0	0	0	0	0	0	0	0	0	α_1
0	0	$\cdots$	0	1	1	...	1	0	0	0	0	0	α_2
$\vdots$		$\cdots$				$\vdots$						$\vdots$	
0	0	$\cdots$	0	0	0	0	0	0	1	1	$\cdots$	1	α_m
1	0	$\cdots$	0	1	0	0	0	0	1	0	0	0	β_1
0	1	$\cdots$	0	0	1	0	0	0	0	1	0	0	β_2
$\vdots$		$\cdots$				$\vdots$						$\vdots$	
0	0	$\cdots$	1	0	0	0	1	0	0	0	0	1	β_n
1	1	$\cdots$	0	1	1	0	0	0	1	1	0	0	γ_1
$\vdots$		$\cdots$				$\vdots$						$\vdots$	
0	0	$\cdots$	1	0	0	0	1	0	0	0	0	1	γ_p
1	1	$\cdots$	0	1	1	0	0	0	1	1	0	0	δ_1
$\vdots$		$\cdots$				$\vdots$						$\vdots$	
0	0	$\cdots$	1	0	0	0	1	0	0	0	0	1	δ_q

TABLE 1.1 – Tableau de transport associé au PT4

1.2.2 Condition de réalisabilité et critère d'optimalité

Théorème 1.2.1 (Condition de réalisabilité [47]). *Le problème de transport à quatre indices admet une solution réalisable si et seulement si*

$$\sum_{i=1}^{m} \alpha_i = \sum_{j=1}^{n} \beta_j = \sum_{k=1}^{p} \gamma_k = \sum_{l=1}^{q} \delta_l. \tag{1.3}$$

Théorème 1.2.2 (Critère d'optimalité [47]). *Soit x une solution réalisable pour le problème (TP4), alors x est optimale si et seulement s'il existe un vecteur*

$$(u_1, \ldots, u_m, v_1, \ldots, v_n, w_1, \ldots, w_p, t_1, \ldots, t_q)^T \in \mathbb{R}^M$$

tel que :

$$u_i + v_j + w_k + t_l = c_{ijkl} \ \textit{pour} \ x_{ijkl} = 0$$

et

$$u_i + v_j + w_k + t_l = c_{ijkl} \ \textit{pour} \ x_{ijkl} > 0.$$

1.2.3 Résolution du problème de transport à quatre indices

Dans cette section, on présente les méthodes de résolution permettant d'obtenir la solution optimale pour un problème de transport à quatre indices.

Algorithme AL_{PT4}

L'algorithme AL_{PT4} est la version sans capacité de AL_{PT4C} introduit par R. Zitouni et A. Keraghel (2003) [44] pour résoudre le problème de transport à quatre indices avec capacité; Il est composé de deux phases :

Phase 1 : Détermination d'une solution de base réalisable initiale

Cette phase permet de déterminer une solution de base réalisable initiale ou confirmer que le $PT4$ est non réalisable.

Pour tout (i,j,k,l), prendre $\hat{\alpha}_i = \alpha_i, \hat{\beta}_j = \beta_j, \hat{\gamma}_k = \gamma_k, \hat{\delta}_l = \delta_l$ et $b_{ijkl} = 0$ (b_{ijkl} est une variable booléenne qui vaut 1 si x_{ijkl} est déjà déterminée) et 0 dans le cas contraire et $E = \{(i,j,k,l),$ tel que $b_{ijkl} = 0\}$.

Tant que $E \neq \varnothing$ **faire**

- Choisir un 4-uplet $(\bar{i},\bar{j},\bar{k},\bar{l}) \in E$ tel que $c_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min\limits_{(i,j,k,l)\in E} c_{ijkl}$
- Prendre $x_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min\left(\hat{\alpha}_{\bar{i}}, \hat{\beta}_{\bar{j}}, \hat{\gamma}_{\bar{k}}, \hat{\delta}_{\bar{l}}\right)$, et $b_{\bar{i}\bar{j}\bar{k}\bar{l}} = 1$ (i.e. $x_{\bar{i}\bar{j}\bar{k}\bar{l}}$ est déterminée)
- Actualiser $\hat{\alpha}_{\bar{i}}, \hat{\beta}_{\bar{j}}, \hat{\gamma}_{\bar{k}}$ et $\hat{\delta}_{\bar{l}}$ comme suit :
  1. $\hat{\alpha}_{\bar{i}} = \hat{\alpha}_{\bar{i}} - x_{\bar{i}\bar{j}\bar{k}\bar{l}}$,
     si $\hat{\alpha}_{\bar{i}} = 0$ alors, prendre $x_{\bar{i}jkl} = 0$ et $b_{\bar{i}jkl} = 1$ pour tout $(j,k,l) \neq (\bar{j},\bar{k},\bar{l})$
  2. $\hat{\beta}_{\bar{j}} = \hat{\beta}_{\bar{j}} - x_{\bar{i}\bar{j}\bar{k}\bar{l}}$,
     si $\hat{\beta}_{\bar{j}} = 0$ alors, prendre $x_{i\bar{j}kl} = 0$ et $b_{i\bar{j}kl} = 1$ pour tout $(i,k,l) \neq (\bar{i},\bar{k},\bar{l})$
  3. $\hat{\gamma}_{\bar{k}} = \hat{\gamma}_{\bar{k}} - x_{\bar{i}\bar{j}\bar{k}\bar{l}}$,
     si $\hat{\gamma}_{\bar{k}} = 0$ alors, prendre $x_{ij\bar{k}l} = 0$ et $b_{ij\bar{k}l} = 1$ pour tout $(i,j,l) \neq (\bar{i},\bar{j},\bar{l})$
  4. $\hat{\delta}_{\bar{l}} = \hat{\delta}_{\bar{l}} - x_{\bar{i}\bar{j}\bar{k}\bar{l}}$,
     si $\hat{\delta}_{\bar{l}} = 0$ alors, prendre $x_{ijk\bar{l}} = 0$ et $b_{ijk\bar{l}} = 1$ pour tout $(i,j,k) \neq (\bar{i},\bar{j},\bar{k})$

Fin tant que.

Phase 2 : Amélioration d'une solution de base réalisable

Elle permet de vérifier l'optimalité d'une solution ou l'améliorer jusqu'à l'optimalité.

Au début de la phase 2, on connaît une solution de base réalisable initiale $x^{(0)}$.

a) Initialisation : Prender $r = 0$,

b) Déterminer l'ensemble $I^{(r)} = \left\{(i,j,k,l)/x_{ijkl}^{(r)} : \text{variable de base}\right\}$,

c) Pour tout $(i,j,k,l) \in I^{(r)}$, résoudre le système linéaire

$$u_i^{(r)} + v_j^{(r)} + w_k^{(r)} + t_l^{(r)} = c_{ijkl},$$

d) Pour tout $(i,j,k,l) \notin I^{(r)}$ déterminer

$$\Delta_{ijkl}^{(r)} = c_{ijkl} - (u_i^{(r)} + v_j^{(r)} + w_k^{(r)} + t_l^{(r)}),$$

Si $\forall (i,j,k,l) \notin I^{(r)}$, on a

$$\Delta_{ijkl}^{(r)} \geq 0$$

alors, la solution $x^{(r)}$ est **optimale**. **Stop**.

Sinon

e) Prendre

$$\Delta_{i_0j_0k_0l_0}^{(r)} = \min_{\Delta_{ijkl}^{(r)}<0} \Delta_{ijkl}^{(r)}$$

f) Pour tout $(i,j,k,l) \in I^{(r)}$, déterminer un cycle $\mu^{(r)}$, en résolvant le système

$$\sum \lambda_{ijkl}^{(r)} P_{ijkl} = -P_{i_0j_0k_0l_0}$$

g) Déterminer $\theta^{(r)}$ avec $\theta^{(r)} = \min\left(\frac{x_{ijkl}^{(r)}}{-\lambda_{ijkl}^{(r)}}\right)$ tel que $\lambda_{ijkl}^{(r)} < 0$

h) Déterminer la nouvelle solution de base $x_{ijkl}^{(r+1)}$ comme suit :

$$x^{(r+1)} = \left\{x_{ijkl}^{(r)}/(i,j,k,l) \notin \mu^{(r)}\right\} \cup \left\{x_{ijkl}^{(r)} + \lambda_{ijkl}^{(r)}\theta^{(r)}/(i,j,k,l) \in \mu^{(r)}\right\}$$

Répéter les étapes **b)**...**h)**.

Méthodes de programmation linéaire

Comme on a vu dans la première section, le problème de transport à quatre indices peut être résolu aussi par des méthodes de la programmation linéaire ; la méthode de simplexe et les méthodes de points intérieurs. Les travaux dans [1,43] montrent que l'algorithme AL_{PT4} est le plus performant pour résoudre un PT4.

1.3 Préliminaires flous

1.3.1 Logique floue

La logique floue est une extension de la logique booléenne crée par Lotfi Zadeh en 1965. En introduisant la notion de degré dans la vérification d'une condition, permettant ainsi à une condition d'être dans un autre état que vrai ou faux, la logique floue confère une flexibilité très appréciable aux raisonnements qui l'utilisent, ce qui rend possible la prise en compte des imprécisions et des incertitudes. Il s'agit d'une approche basée sur le raisonnement humain plutôt que sur des calculs rigides. Pour des problèmes mal définis, la pensée humaine étant irremplaçable.

En effet, le mode de raisonnement en logique floue est plus intuitif que la logique classique. Il permet aux concepteurs de mieux appréhender les phénomènes naturels, imprécis et difficilement modélisables en s'appuyant sur quelques notions telles que les fonctions d'appartenance à des ensembles dits "ensembles flous" et les fonctions de classement des nombres dits "nombres flous".

Applications de la logique floue

La logique floue trouve ses applications dans de nombreux domaines, comme modèle de représentation et de manipulation des connaissances. Elle est particulièrement utilisée pour la mise au point de contrôleurs et des systèmes experts flous, pour l'aide à la décision, pour l'interrogation flexible des bases de données et dans le filtrage flou. En 1975, le professeur Mamdani à Londres [26] a développé une stratégie pour le contrôle des procédés et présente les résultats très encourageants qu'il a obtenus sur la conduite d'un moteur à vapeur. En 1978, la société danoise F.L. Smidth & Co réalise le contrôle d'un four à ciment, c'est la première véritable application industrielle de la logique floue.

1.3.2 Notions floues

Sous-ensemble flou

La logique floue repose sur la théorie des ensembles flous, qui est une généralisation de la théorie des ensembles classiques. Un sous-ensemble flou est complètement défini par la donnée de sa fonction d'appartenance. A partir de cette fonction, un certain nombre de caractéristiques du sous-ensemble flou peuvent être étudiées.

Fonction d'appartenance

Au lieu d'imposer l'appartenance d'un élément à l'ensemble "vrai" ou l'ensemble "faux" comme dans la logique binaire traditionnelle, la logique floue admet des degrés d'appartenance. Le degré d'appartenance d'un élément à un ensemble flou est matérialisé par un nombre compris entre 0 et 1.

Avec un ensemble U, on associe une fonction $\mu : U \to [0, 1]$ appelée fonction d'appartenance. Étant donné $x \in U$, la valeur $\mu(x)$ est appelée "facteur d'appartenance".

Notons que cette fonction d'appartenance est l'équivalent de la fonction caractéristique d'un ensemble classique.

Ensemble flou

Un ensemble flou $\tilde{A}$ est défini comme l'ensemble des couples $(x, \mu_{\tilde{A}}(x))$, où x est un élément de l'univers du discours U et $\mu_{\tilde{A}}$ est la fonction d'appartenance.

L'interprétation de cette fonction est la suivante :

— $\mu_{\tilde{A}(x)} = 1$ signifie que x appartient à $\tilde{A}$ avec certitude.

— $\mu_{\tilde{A}(x)} = 0$ signifie que x n'appartient pas à $\tilde{A}$ avec certitude.

— Les autres valeurs signifient le degré d'appartenance de x à $\tilde{A}$.

Example 1.1. La jeunesse d'une personne d'âge x ans peut être caractérisée par l'ensemble flou $\tilde{A}$ dont la fonction d'appartenance est par exemple :

$$\mu_{\tilde{A}}(x) = \begin{cases} 1 & \text{si} \quad 0 \leq x \leq 25, \\ \frac{40-x}{15} & \text{si} \quad 25 \leq x \leq 40, \\ 0 & \text{ailleurs.} \end{cases}$$

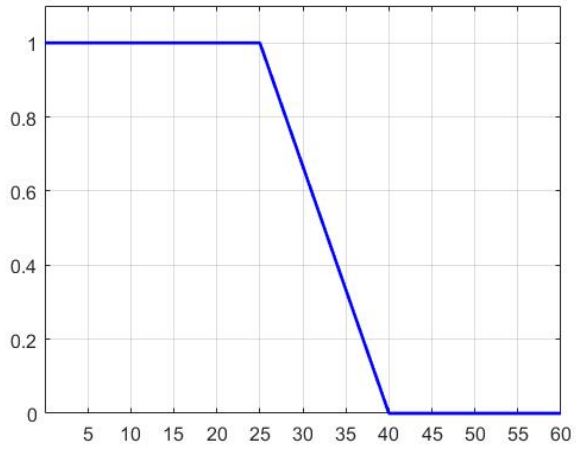

FIGURE 1.1 – Représentation du nombre flou de cet exemple

Caractéristiques d'un sous-ensemble flou

Soit U un ensemble, $\tilde{A}$ un sous-ensemble flou de U et $\mu_{\tilde{A}}$ sa fonction d'appartenance.

- **Support de $\tilde{A}$:** c'est l'ensemble des éléments de U qui appartiennent au moins un peu à $\tilde{A}$. Il est défini par
$$supp(\tilde{A}) = \{x \in U / \mu_{\tilde{A}}(x) > 0\}.$$
Un sous-ensemble flou $\tilde{A}$ dont le support est un singleton dans U avec $\mu_{\tilde{A}}(x) = 1$ est appelé " **Singleton flou** ".
- **Hauteur de $\tilde{A}$:** c'est sa plus grande valeur prise par sa fonction d'appartenance. Elle est définie par :
$$h(\tilde{A}) = \sup_{x \in U}(\mu_{\tilde{A}}(x)).$$
- **Noyau de $\tilde{A}$:** c'est l'ensemble des éléments de U pour lesquels la fonction d'appartenance de $\tilde{A}$ vaut 1. Il est défini par :
$$noy(\tilde{A}) = \{x \in U / \mu_{\tilde{A}}(x) = 1\}.$$

Lorsque le noyau est réduit à un point, celui-ci est appelé "valeur modale".

Opérations sur les sous-ensembles flous

Supposons que $\tilde{A}$ et $\tilde{B}$ sont deux sous-ensembles flous définis dans un univers du discours U par les fonctions d'appartenance $\mu_{\tilde{A}}$ et $\mu_{\tilde{B}}$. On peut définir des opérations ensemblistes telles que : égalité, inclusion, intersection, union et complément grâce à des opérations sur les fonctions d'appartenance.

- **Égalité :** $\tilde{A} = \tilde{B}$ ssi $\mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) \quad \forall x \in U$.
- **Inclusion :** $\tilde{A}$ est dit inclus dans $\tilde{B}$, propriété que l'on note $\tilde{A} \subseteq \tilde{B}$, si tout élément x de U qui appartient à $\tilde{A}$ appartient aussi à $\tilde{B}$ avec un degré au moins aussi grand :
$$\mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x) \quad \forall\, x \in U.$$
- **Intersection :** L'intersection de $\tilde{A}$ et $\tilde{B}$, noté $\tilde{A} \cap \tilde{B}$, est le sous-ensemble flou constitué des éléments x de U affectés du plus petit des deux degrés d'appartenance $\mu_{\tilde{A}}$ et $\mu_{\tilde{B}}$:
$$\mu_{\tilde{A} \cap \tilde{B}}(x) = \min(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) \quad \forall\, x \in U.$$
- **Union :** L'union de $\tilde{A}$ et $\tilde{B}$, noté $\tilde{A} \cup \tilde{B}$, est le sous-ensemble flou constitué des éléments de U affectés du plus grand des deux degrés d'appartenance $\mu_{\tilde{A}}$ et $\mu_{\tilde{B}}$:
$$\mu_{\tilde{A} \cup \tilde{B}}(x) = \max(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) \quad \forall x \in U.$$

- **Complément :** Le complément de $\tilde{A}$, que l'on note $\tilde{A}^c$, est le sous-ensemble flou de U constitué des éléments x lui appartenant d'autant plus qu'ils appartiennent peu à $\tilde{A}$:

$$\mu_{\tilde{A}^c} = 1 - \mu_{\tilde{A}}(x) \quad \forall x \in U.$$

Propriétés

Soient $\tilde{A}$ et $\tilde{B}$ deux sous-ensembles flous de U, on a :

- **Associativité :**

$$\begin{aligned} \tilde{A} \cap (\tilde{B} \cap \tilde{C}) &= (\tilde{A} \cap \tilde{B}) \cap \tilde{C}, \\ \tilde{A} \cup (\tilde{B} \cup \tilde{C}) &= (\tilde{A} \cup \tilde{B}) \cup \tilde{C}. \end{aligned}$$

- **Commutativité :**

$$\begin{aligned} \tilde{A} \cap \tilde{B} &= \tilde{B} \cap \tilde{A}, \\ \tilde{A} \cup \tilde{B} &= \tilde{B} \cup \tilde{A}. \end{aligned}$$

- **Distributivités mutuelles :**

$$\begin{aligned} \tilde{A} \cap (\tilde{B} \cup \tilde{C}) &= (\tilde{A} \cap \tilde{B}) \cup (\tilde{A} \cap \tilde{C}), \\ \tilde{A} \cup (\tilde{B} \cap \tilde{C}) &= (\tilde{A} \cup \tilde{B}) \cap (\tilde{A} \cup \tilde{C}). \end{aligned}$$

Nombre flou

Un nombre réel flou $\tilde{A}$ est un sous-ensemble flou de l'ensemble des nombres réels $\mathbb{R}$ avec une fonction d'appartenance $\mu_{\tilde{A}}$.

Nombre flou trapézoïdal :

On appelle nombre flou trapézoïdal, un nombre flou $\tilde{A} = (a_1, a_2, a_3, a_4)$ avec $a_1 \leq a_2 \leq a_3 \leq a_4$ et de fonction d'appartenance

$$\mu_{\tilde{A}}(x) = \begin{cases} 0 & \text{si} \quad x \leq a_1, \\ \frac{x-a_1}{a_2-a_1} & \text{si} \quad a_1 \leq x \leq a_2, \\ 1 & \text{si} \quad a_2 \leq x \leq a_3, \\ \frac{a_4-x}{a_4-a_3} & \text{si} \quad a_3 \leq x \leq a_4, \\ 0 & \text{si} \quad a_4 \leq x. \end{cases}$$

La fonction d'appartenance $\mu_{\tilde{A}}$ est continue par morceaux sur $\mathbb{R}$, continue sur $[\,0,1\,]$, strictement croissante sur $[\,a_1,a_2\,]$, constante sur $[\,a_2,a_3\,]$ et strictement décroissante sur $[\,a_3,a_4\,]$.

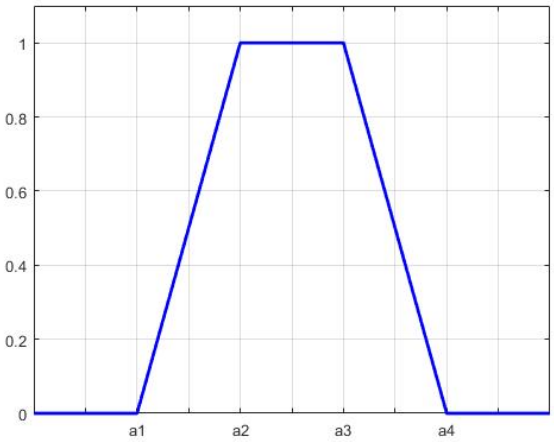

FIGURE 1.2 – Fonctions d'appartenance d'un nombre flou trapézoïdal

Nombre flou triangulaire :

Si $a_2 = a_3$ dans la représentation (a_1,a_2,a_3,a_4) le nombre flou est appelé alors, nombre flou triangulaire, c'est l'un des nombres flous les plus utilisés. Sa fonction d'appartenance est donnée par :

$$\mu_{\tilde{A}}(x) = \begin{cases} 0 & \text{si} \quad x \leq a_1, \\ \frac{x-a_1}{a_2-a_1} & \text{si} \quad a_1 \leq x \leq a_2, \\ \frac{a_4-x}{a_4-a_2} & \text{si} \quad a_2 \leq x \leq a_4, \\ 0 & \text{si} \quad a_4 \leq x. \end{cases}$$

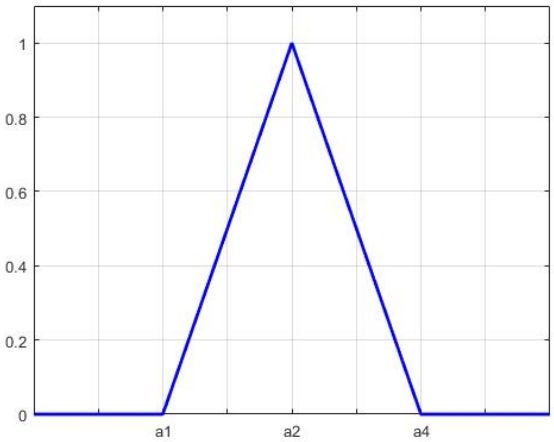

FIGURE 1.3 – Fonctions d'appartenance d'un nombre flou triangulaire

Fonction de classement

Appelons $F(\mathbb{R})$ l'ensemble des nombres flous définis sur $\mathbb{R}$. On appelle fonction de classement, une fonction $\Re : F(\mathbb{R}) \to \mathbb{R}$ permettant de comparer les nombres flous.

Par exemple,

- Si $F(\mathbb{R})$ représente l'ensemble des nombres flous trapézoïdaux,

$$\Re(\tilde{A}) = \frac{a_1 + a_2 + a_3 + a_4}{4}, \quad \tilde{A} = (a_1, a_2, a_3, a_4).$$

- Si $F(\mathbb{R})$ représente l'ensemble des nombres flous triangulaires donc :

$$\Re(\tilde{A}) = \frac{a_1 + 2a_2 + a_4}{4}, \quad \tilde{A} = (a_1, a_2, a_4).$$

Pour deux nombres flous trapézoïdaux $\tilde{A} = (a_1, a_2, a_3, a_4)$ et $\tilde{B} = (b_1, b_2, b_3, b_4)$, nous avons

$$\tilde{A} \leq_{\Re} \tilde{B} \iff \Re(\tilde{A}) \leq \Re(\tilde{B}),$$

$$\tilde{A} \geq_{\Re} \tilde{B} \iff \Re(\tilde{A}) \geq \Re(\tilde{B}),$$

$$\tilde{A} =_{\Re} \tilde{B} \iff \Re(\tilde{A}) = \Re(\tilde{B}).$$

Opérations arithmétiques

Soient $\tilde{A} = (a_1, a_2, a_3, a_4)$ et $\tilde{B} = (b_1, b_2, b_3, b_4)$ deux nombres flous trapézoïdaux :

- **Addition :** $\tilde{A} \oplus \tilde{B} = (a_1, a_2, a_3, a_4) \oplus (b_1, b_2, b_3, b_4) = (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_4 + b_4)$.
- **Soustraction :** $\tilde{A} \ominus \tilde{B} = (a_1, a_2, a_3, a_4) \ominus (b_1, b_2, b_3, b_4) = (a_1 - b_4, a_2 - b_3, a_3 - b_2, a_4 - b_1)$.
- **Multiplication :**

  — Lorsque $\Re(\tilde{B}) > 0$

  $$\tilde{A} \otimes \tilde{B} = \Re(\tilde{B})\,(a_1, a_2, a_3, a_4)\,,$$

  — Et lorsque $\Re(\tilde{B}) < 0$

  $$\tilde{A} \otimes \tilde{B} = \Re(\tilde{B})\,(a_4, a_3, a_2, a_1)\,.$$

Propriétés

Soient $\tilde{A}$ et $\tilde{B}$ deux nombres flous

- L'addition des nombres flous est associative et commutative :

$$\tilde{A} \oplus \tilde{B} = \tilde{B} \oplus \tilde{A} \text{ et } \tilde{A} \oplus \left(\tilde{B} \oplus \tilde{Z}\right) = \left(\tilde{A} \oplus \tilde{B}\right) \oplus \tilde{Z} \quad \forall \tilde{A}, \tilde{B}, \tilde{Z} \in F(\mathbb{R})$$

- Aucun élément $\tilde{A} \in F(\mathbb{R})$ possède un opposé dans $F(\mathbb{R})$.
- La multiplication d'un nombre flou $\tilde{A} = (a_1, a_2, a_3, a_4)$ par un nombre réel $\alpha > 0$ est définie par : $\alpha\tilde{A} = (\alpha a_1, \alpha a_2, \alpha a_3, \alpha a_4)$.
- La multiplication d'un nombre flou $\tilde{A} = (a_1, a_2, a_3, a_4)$ par un nombre réel $\alpha < 0$ est définie par : $\alpha\tilde{A} = (\alpha a_4, \alpha a_3, \alpha a_2, \alpha a_1)$.
- $(\alpha + \beta)\ \tilde{A} = \alpha\ \tilde{A} \oplus \beta\ \tilde{A}$, $\forall \alpha, \beta \in \mathbb{R}$ avec $\alpha\beta \geq 0$ et $\forall \tilde{A} \in F(\mathbb{R})$.
- $\alpha(\tilde{A} \oplus \tilde{B}) = \alpha\tilde{A} \oplus \alpha\tilde{B}$, $\forall \alpha \in \mathbb{R}$ et $\forall\ \tilde{A}, \tilde{B} \in F(\mathbb{R})$.

CHAPITRE 2

PROBLÈME DE TRANSPORT FLOU À QUATRE INDICES

2.1 Introduction

La théorie floue est presque devenue une mode pendant les années 90. Beaucoup de chercheurs dans différents domaines scientifiques utilisent la théorie formulée par L. Zadeh [41]. Elle a connu un intérêt important dans la communauté scientifique au cours des dernières années.

Elle apparaît dans plusieurs domaines tels que : l'aide à la décision, au diagnostic (domaine médical, orientation professionnelle,...), base de données (objets flous et/ou requêtes floues), optimisation, commande floue de systèmes, ..., etc.

Dans ces dernières années, l'attention des chercheurs de ce domaine s'est concentrée sur la résolution du problème de transport dans un environnement flou où les données du problème sont complètement ou partiellement présentées par des nombres flous. Il s'agit d'une application plus cohérente du problème de transport réel, en raison de l'outil de mesure imprécis, des données perdues et manquantes, et même des erreurs de calcul. Le problème de transport flou (PTF) peut être modélisé en trois types.

1. Type 1 : Il s'agit du cas où les demandes et les offres du problème sont incertaines (floues), mais les coûts sont nets (par opposé au mot "flou").
2. Type 2 : Il s'agit du cas où les demandes et les offres du problème sont nettes, mais les coûts sont incertains (flous).
3. Type 3 : Il s'agit du cas où toutes les données sont floues.

2.2 Etat de l'art

Dans la littérature, de nombreux chercheurs ont développé des algorithmes efficaces pour résoudre le PTF dans tous ces différents types. Zimmermann (1978) [42] a montré que les algorithmes de programmation linéaire floue étaient efficaces. Ohegeartaigh (1982) [27] a proposé un algorithme pour résoudre le problème de transport avec des offres floues et des demandes floues (type-1). Il montre également qu'il existe une relation évidente entre la formulation de

programmes linéaires nets et leur équivalent flou. Chanas et al. (1984) [8] ont analysé le problème du transport avec des offres floues et des demandes floues. Ils ont utilisé la technique de programmation paramétrique qui leur permet d'obtenir non seulement la solution optimale, mais aussi d'autres solutions alternatives qui s'en rapprochent. Chanas et Kuchta (1996) [9] ont introduit le problème du transport avec des coefficients de coût flous et des valeurs nettes de l'offre et de la demande (type-2). Ils ont défini la solution optimale de ce problème et proposé un algorithme pour déterminer cette solution. Kaur et Kumar (2011) [23] ont proposé une nouvelle méthode qui est une extension directe de la méthode classique pour résoudre le problème de transport totalement flou. Ils prennent les données du problème sous forme de nombres flous trapézoïdaux généralisés. Un nouvel algorithme pour résoudre le problème de transport flou a été proposé par Kaur et Kumar (2012) [20]. Ils ont d'abord adapté trois méthodes pour trouver une solution de base réalisable initiale, puis ils ont généralisé la méthode de distribution modifiée floue pour déterminer la solution optimale. Dans l'algorithme proposé ; les coûts de transport sont représentés par des nombres flous trapézoïdaux généralisés, mais l'offre et la demande sont des quantités nettes. Singh. S. K., Yadav (2014) [37] ont proposé une méthode floue intuitionniste pour trouver une solution de base réalisable en termes de nombres flous intuitionnistes triangulaires où le problème de transport présente une incertitude et une hésitation dans l'offre et la demande. Ensuite, ils ont proposé une méthode de distribution modifiée floue intuitionniste pour trouver une solution optimale. Les différents types de problème de transport flou ont été évoqués par P. Senthil Kumar (2016) [33, 34]. Il a proposé une nouvelle méthode appelée méthode PSK pour résoudre chaque type. Ebrahimnejad, A., Verdegay, JL (2017) [12] ont formulé le problème de transport flou intuitionniste (IFTP) et ont proposé une approche qui fournit une solution optimale floue intuitionniste pour résoudre ce problème.

Le problème de transport flou à quatre indices (PTF4) est une généralisation du PTF classique. Le choix de plus de deux indices (offre, demande) dans le problème de transport nous permet de nous donner une idée pour le cas plus général du problème de transport à indices multiples. Récemment, M. Hedid et al. (2020) [16, 17] ont introduit des approches obtenues par l'adaptation de quelques méthodes classiques pour résoudre le problème de transport flou à quatre indices.

2.3 Position du problème

2.3.1 Interprétation économique

Soient :

- $A_1, \ldots, A_m$, m origines de disponibilités $\tilde{\alpha}_1, \ldots, \tilde{\alpha}_m$, respectivement.
- $B_1, \ldots, B_n$, n destinations de demandes $\tilde{\beta}_1, \ldots, \tilde{\beta}_n$, respectivement.
- $S_1, \ldots, S_p$, p moyens de transport choisis convenablement de charges réservées $\tilde{\gamma}_1, \ldots, \tilde{\gamma}_p$,

respectivement.

- $H_1, \ldots, H_q$, q qualités des marchandises prises en même unités, de quantité $\tilde{\delta}_1, \ldots, \tilde{\delta}_q$, respectivement.
- $\tilde{x}_{ijkl}$ $(i = 1, \ldots, m, j = 1, \ldots, n, k = 1, \ldots, p, l = 1, \ldots, q)$: la quantité du produit H_l transportée de l'origine A_i vers la destination B_j par le moyen de transport S_k.
- $\tilde{c}_{ijkl}$ $(i = 1, \ldots, m, j = 1, \ldots, n, k = 1, \ldots, p, l = 1, \ldots, q)$: le coût unitaire de transport d'un produit $\tilde{x}_{ijkl}$.

2.3.2 Formulation mathématique

Le problème de transport flou à quatre indices peut être formulé mathématiquement en trois types :

1. **Type 1** : Dans ce type, les données (offres, demandes, moyens de transport et qualités des produits) sont présentées par des nombres flous, mais les coûts de transport sont présentés par des nombres nets. Sa formulation est :

$$\text{Minimiser } \tilde{Z} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} c_{ijkl}\tilde{x}_{ijkl}$$

sous les contraintes :

$$\begin{aligned}
\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{x}_{ijkl} &=_{\Re} \tilde{\alpha}_i \text{ pour tout } \quad i = 1, \ldots, m,\\
\sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{x}_{ijkl} &=_{\Re} \tilde{\beta}_j \text{ pour tout } \quad j = 1, \ldots, n,\\
\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q}\tilde{x}_{ijkl} &=_{\Re} \tilde{\gamma}_k \text{ pour tout } \quad k = 1, \ldots, p,\\
\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\tilde{x}_{ijkl} &=_{\Re} \tilde{\delta}_l \text{ pour tout } \quad l = 1, \ldots, q,\\
\tilde{x}_{ijkl} &\geq_{\Re} 0, \text{ pour tout } i = 1 : m, j = 1 : n, k = 1 : p, l = 1 : q.
\end{aligned}$$

Avec $\tilde{\alpha}_i >_{\Re} 0$, $\tilde{\beta}_j >_{\Re} 0$, $\tilde{\gamma}_k >_{\Re} 0$, $\tilde{\delta}_l >_{\Re} 0$ et $c_{ijkl} \geq 0$, $\forall(i, j, k, l)$.

2. **Type 2** : Dans ce type, les données (offres, demandes, moyens de transport et qualités des produits) sont présentées par des nombres nets, mais les coûts de transport sont présentés par des nombres flous. Sa formulation est :

$$\text{Minimiser } \tilde{Z} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{c}_{ijkl}x_{ijkl}$$

sous les contraintes :

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}x_{ijkl} = \alpha_i \text{ pour tout } \quad i = 1,\dots,m,\\
&\sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q}x_{ijkl} = \beta_j \text{ pour tout } \quad j = 1,\dots,n,\\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q}x_{ijkl} = \gamma_k \text{ pour tout } \quad k = 1,\dots,p,\\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}x_{ijkl} = \delta_l \text{ pour tout } l = 1,\dots,q,\\
&\qquad x_{ijkl} \geq 0, \text{ pour tout } i = 1:m, j = 1:n, k = 1:p, l = 1:q.
\end{aligned}$$

Avec $\alpha_i > 0$, $\beta_j > 0$, $\gamma_k > 0$, $\delta_l > 0$ et $\tilde{c}_{ijkl} \geq_{\Re} 0$, $\forall(i,j,k,l)$.

3. **Type 3 (totalement flou)** : Dans ce type, toutes les data du problème sont des nombres flous. Sa formulation est :

$$\text{Minimiser } \tilde{Z} =_{\Re} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{c}_{ijkl} \otimes \tilde{x}_{ijkl}$$

sous les contraintes :

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{x}_{ijkl} =_{\Re} \tilde{\alpha}_i \text{ pour tout } \quad i = 1,\dots,m,\\
&\sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{x}_{ijkl} =_{\Re} \tilde{\beta}_j \text{ pour tout } \quad j = 1,\dots,n,\\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q}\tilde{x}_{ijkl} =_{\Re} \tilde{\gamma}_k \text{ pour tout } \quad k = 1,\dots,p,\\
&\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\tilde{x}_{ijkl} =_{\Re} \tilde{\delta}_l \text{ pour tout } \quad l = 1,\dots,q,\\
&\qquad \tilde{x}_{ijkl} \geq_{\Re} 0, \text{ pour tout } i = 1:m, j = 1:n, k = 1:p, l = 1:q.
\end{aligned}$$

Avec $\tilde{\alpha}_i >_{\Re} 0$, $\tilde{\beta}_j >_{\Re} 0$, $\tilde{\gamma}_k >_{\Re} 0$, $\tilde{\delta}_l >_{\Re} 0$, et $\tilde{c}_{ijkl} \geq_{\Re} 0$, $\forall(i,j,k,l)$.

Le problème de transport flou à quatre indices peut être écrit aussi sous forme d'un programme linéaire flou :

$$\begin{cases}
\min \tilde{Z} =_{\Re} \tilde{c}^T \otimes \tilde{x}\\
\text{sous les contraintes :}\\
A\tilde{x} =_{\Re} \tilde{b}\\
\Re(\tilde{x}) \geq 0
\end{cases} \tag{2.1}$$

Où :
— $\tilde{x} = (\tilde{x}_{1111}, \ldots, \tilde{x}_{mnpq})^T \in F(\mathbb{R})^N$.
— $\tilde{c} = (\tilde{c}_{1111}, \ldots, \tilde{c}_{mnpq})^T \in F(\mathbb{R})^N$.
— $\tilde{b} = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_m, \tilde{\beta}_1, \ldots, \tilde{\beta}_n, \tilde{\gamma}_1, \ldots, \tilde{\gamma}_p, \tilde{\delta}_1, \ldots, \tilde{\delta}_q) \in F(\mathbb{R})^M$.
— A est une $M \times N$ matrice.
— $M = m + n + p + q$ et $N = mnpq$.

Soit $E = \{(i, j, k, l);\ i = 1 : m, j = 1 : n, k = 1 : p$ et $l = 1 : q\}$ à chaque $(i, j, k, l) \in E$, on associe un vecteur $P_{ijkl} \in \mathbb{R}^M$. Les composants de ce vecteur sont des zéros sauf quatre ; ils sont situés dans les lignes $i, m + j, m + n + k$, et $m + n + p + l$ et ayant 1 comme valeur commune. On définit la matrice A comme la matrice des vecteurs P_{ijkl}. Notons que $rang(A) = M - 3$.

Le problème est de déterminer $\tilde{x}_{ijkl}$ pour que le coût total du transport soit minimal.

Définition 2.1. Une solution réalisable $\tilde{x}$ d'un PTF4 est dite de base si les colonnes de la matrice $A_{\tilde{x}}$ obtenue de A en gardant seulement les colonnes correspondantes aux variables $\tilde{x}_{ijkl} >_{\Re} 0$ sont linéairement indépendantes.

Définition 2.2. Soit $\tilde{x}$ une solution de base réalisable pour un PTF4, le quadruplet (i, j, k, l) tel que $\tilde{x}_{ijkl} >_{\Re} 0$ est appelé case intéressante.

2.4 Résolution du PTF4

2.4.1 Condition de réalisabilité

Une condition nécessaire pour que le problème de transport flou à quatre indices possède une solution réalisable est :

Théoreme 2.1. *Le problème de transport flou à quatre indices de type* 1 *admet une solution réalisable si seulement si :*

$$\sum_{i=1}^{m} \tilde{\alpha}_i =_{\Re} \sum_{j=1}^{n} \tilde{\beta}_j =_{\Re} \sum_{k=1}^{p} \tilde{\gamma}_k =_{\Re} \sum_{l=1}^{q} \tilde{\delta}_l =_{\Re} H.$$

Théoreme 2.2. *Le problème de transport flou à quatre indices de type* 2 *admet une solution réalisable si seulement si :*

$$\sum_{i=1}^{m} \alpha_i = \sum_{j=1}^{n} \beta_j = \sum_{k=1}^{p} \gamma_k = \sum_{l=1}^{q} \delta_l = H.$$

Théoreme 2.3. *Le problème de transport flou à quatre indices de type* 3 *admet une solution réalisable si seulement si :*

$$\sum_{i=1}^{m} \tilde{\alpha}_i =_{\Re} \sum_{j=1}^{n} \tilde{\beta}_j =_{\Re} \sum_{k=1}^{p} \tilde{\gamma}_k =_{\Re} \sum_{l=1}^{q} \tilde{\delta}_l =_{\Re} H.$$

Remarque 2.1. Les Théorèmes 2.1,2.2 et 2.3 sont des généralisations du Théorème 1.2.1.

Pour obtenir la solution optimale floue d'un problème de transport flou à quatre indices, nous passons par deux phases :

1. Détermination d'une solution de base réalisable initiale.
2. Amélioration d'une solution de base réalisable.

2.4.2 Phase 1

Les méthodes de résolution permettant d'obtenir une solution de base réalisable initiale pour un problème de transport à deux indices peuvent être généralisées pour un tel problème à quatre indices dans un environnement flou.

On va présenter une adaptation des méthodes de coût minimum, d'approximation de Russell et d'approximation de Vogel pour un PTF4 dans un contexte flou. On note FLC4, FRAM4, et FVAM4 les algorithmes adaptés respectivement.

Soit E_b l'ensemble des cases intéressantes. Au début de chaque algorithme, on a $E_b = \emptyset$.

Algorithme FLC4

Etape 1 :

1. Pour chaque quadruplet (i, j, k, l) non saturé, choisir $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$, où $\tilde{c}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min \tilde{c}_{ijkl}$.
2. Prendre $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min\left(\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}, \tilde{\delta}_{\bar{l}}\right)$ et ajouter $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$ à E_b.
3. Actualiser $\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}$ et $\tilde{\delta}_{\bar{l}}$ comme suit :
   (a) $\tilde{\alpha}_{\bar{i}} = \tilde{\alpha}_{\bar{i}} \ominus \tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}}$
   Si $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \tilde{\alpha}_{\bar{i}}$ alors prendre $\tilde{x}_{\bar{i}jkl}$ égal à zéro flou et saturer $\tilde{c}_{\bar{i}jkl}$, $\forall (j,k,l) \neq (\bar{j}, \bar{k}, \bar{l})$.
   (b) $\tilde{\beta}_{\bar{j}} = \tilde{\beta}_{\bar{j}} \ominus \tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}}$
   Si $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \tilde{\beta}_{\bar{j}}$ alors prendre $\tilde{x}_{i\bar{j}kl}$ égal à zéro flou et saturer $\tilde{c}_{i\bar{j}kl}$, $\forall (i,k,l) \neq (\bar{i}, \bar{k}, \bar{l})$.
   (c) $\tilde{\gamma}_{\bar{k}} = \tilde{\gamma}_{\bar{k}} \ominus \tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}}$
   Si $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \tilde{\gamma}_{\bar{k}}$ alors prendre $\tilde{x}_{ij\bar{k}l}$ égal à zéro flou et saturer $\tilde{c}_{ij\bar{k}l}$, $\forall (i,j,l) \neq (\bar{i}, \bar{j}, \bar{l})$.
   (d) $\tilde{\delta}_{\bar{l}} = \tilde{\delta}_{\bar{l}} \ominus \tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}}$
   Si $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \tilde{\delta}_{\bar{l}}$ alors prendre $\tilde{x}_{ijk\bar{l}}$ égal à zéro flou et saturer $\tilde{c}_{ijk\bar{l}}$, $\forall (i,j,k) \neq (\bar{i}, \bar{j}, \bar{k})$.

Etape 2 :

- Répéter 1 à 3 jusqu'à toutes les variables $\tilde{x}_{ijkl}$ soient déterminées.

Algorithme FRAM4

Etape 1 :

1. Pour tout i non saturé, déterminer $\tilde{c}^i = \max\limits_{jkl} \tilde{c}_{ijkl}$.
2. Pour tout j non saturé, déterminer $\tilde{c}^j = \max\limits_{ikl} \tilde{c}_{ijkl}$.
3. Pour tout k non saturé, déterminer $\tilde{c}^k = \max\limits_{ijl} \tilde{c}_{ijkl}$.
4. Pour tout l non saturé, déterminer $\tilde{c}^l = \max\limits_{ijk} \tilde{c}_{ijkl}$.
5. Calculer les coûts réduits $c^*_{ijkl} = \tilde{c}_{ijkl} \ominus (\tilde{c}^i \oplus \tilde{c}^j \oplus \tilde{c}^k \oplus \tilde{c}^l)$.
6. Choisir la case $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$ qui correspond au moindre coût réduit c^*_{ijkl} ; s'il y en a plusieurs, choisissez celle qui a le plus petit coût flou $\tilde{c}_{ijkl}$; s'il y a encore égalité, choisir celle dont $\min(\tilde{\alpha}_i, \tilde{\beta}_j, \tilde{\gamma}_k, \tilde{\delta}_l)$ est le plus grand.
7. Prendre $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min\left(\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}, \tilde{\delta}_{\bar{l}}\right)$ et ajouter $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$ à E_b.
8. Actualiser $\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}$ et $\tilde{\delta}_{\bar{l}}$ comme à l'étape 1 (3) dans l'algorithme FLC4.

Etape 2 :

- Répéter 1 à 8 jusqu'à ce que toutes les $\tilde{x}_{ijkl}$ soient déterminées.

Algorithme FVAM4

La pénalité d'une rangé i, j, k ou l est donnée par la différence entre le coût flou le plus petit et le deuxième plus petit.

Etape 1 :

1. Pour tout i non saturé, déterminer la pénalité $p_i^1 = \min_2^i \ominus \min_1^i$.
2. Pour tout j non saturé, déterminer la pénalité $p_j^2 = \min_2^j \ominus \min_1^j$.
3. Pour tout k non saturé, déterminer la pénalité $p_k^3 = \min_2^k \ominus \min_1^k$.
4. Pour tout l non saturé, déterminer la pénalité $p_l^4 = \min_2^l \ominus \min_1^l$.
5. Identifier la rangé correspondante à la plus grande pénalité.
6. Dans la rangé sélectionnée trouvée à l'étape précédente, identifier la case $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$ qui correspond au plus petit coût flou.
7. Prendre $\tilde{x}_{\bar{i}\bar{j}\bar{k}\bar{l}} = \min(\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}, \tilde{\delta}_{\bar{l}})$ et ajouter $(\bar{i}, \bar{j}, \bar{k}, \bar{l})$ à E_b.
8. Actualiser $\tilde{\alpha}_{\bar{i}}, \tilde{\beta}_{\bar{j}}, \tilde{\gamma}_{\bar{k}}$ et $\tilde{\delta}_{\bar{l}}$ comme à l'étape 1 (3) dans l'algorithme FLC4.

Etape 2 :

- Répéter 1 à 8 jusqu'à ce que toutes les $\tilde{x}_{ijkl}$ soient déterminées.

2.4.3 Traitement de dégénérescence

A la fin de la phase 1, on obtient une solution de base réalisable initiale, qui peut être dégénérée.

Soit A_x la matrice des vecteurs $P_{ijkl}/(i,j,k,l) \in E_b$.

Test de la dégénérescence

- **Si** $rang(A_x) = rang(A)$, **alors**
  la solution obtenue est non dégénérée.
- **Si** $rang(A_x) < rang(A)$, **alors**
  la solution obtenue est dégénérée.

Traitement de dégénérescence

Soit E_h le complément de E_b dans l'ensemble E (i.e., $E_h = E_b^c$) et soit E_s un ensemble de s éléments de E_h avec $s = rang(A) - rang(A_x)$.

— Si $E_b \cup E_s$ est libre, alors prendre $I = E_b \cup E_s$,

— Sinon modifier E_s et refaire le même test précédent.

Il s'agit d'une modification de la méthode de [45].

2.4.4 Phase 2

Pour tester l'optimalité ou pour améliorer une solution de base réalisable, nous allons adapter la phase 2 de AL_{PT4C} [44] dans un contexte flou.

1. Initialisation :
   Soit $I^{(r)} = \{(i,j,k,l)$ tel que $\tilde{x}_{ijkl}$ variable de base à l'itération $r\}$. Au début on a $r = 0$ et $I^{(0)} = I$.
2. Pour tout $(i,j,k,l) \in I^{(r)}$, résoudre le système linéaire :

$$A_x Y =_{\Re} \tilde{c}_{ijkl} \quad \text{où} \quad Y = [\tilde{u}_i^{(r)}, \tilde{v}_j^{(r)}, \tilde{w}_k^{(r)}, \tilde{t}_l^{(r)}]^T; i = 1:m, j = 1:n, k = 1:p, l = 1:q$$

3. Pour tout $(i,j,k,l) \notin I^{(r)}$ déterminer

$$\tilde{\Delta}_{ijkl}^{(r)} = \tilde{c}_{ijkl} \ominus (\tilde{u}_i^{(r)} \oplus \tilde{v}_j^{(r)} \oplus \tilde{w}_k^{(r)} \oplus \tilde{t}_l^{(r)})$$

4. - **Si** $\forall (i,j,k,l) \notin I^{(r)}$, on trouve $\Re\left(\tilde{\Delta}_{ijkl}^{(r)}\right) \geq 0$ **alors**
     La solution $\tilde{x}^{(r)}$ est **optimale**. **Stop**.
   - **Sinon**, Prendre

$$\tilde{\Delta}_{i_0 j_0 k_0 l_0}^{(r)} = \min\{\tilde{\Delta}_{ijkl}^{(r)};\ \Re\left(\tilde{\Delta}_{ijkl}^{(r)}\right) < 0\}.$$

(a) Pour tout $(i,j,k,l) \in I^{(r)}$, déterminer le cycle $\mu^{(r)}$ en résolvant le système :

$$\sum \lambda_{ijkl}^{(r)} A_x = -P_{i_0 j_0 k_0 l_0}$$

(b) Déterminer $\theta^{(r)}$ où $\theta^{(r)} = \min\ (\frac{\tilde{x}_{ijkl}^{(r)}}{-\lambda_{ijkl}^{(r)}})$avec $\lambda_{ijkl}^{(r)} < 0$

(c) Déterminer la nouvelle solution de base $\tilde{x}^{(r+1)}$ comme suit :

$$\tilde{x}^{(r+1)} = \left\{ \tilde{x}_{ijkl}^{(r)} / (i,j,k,l) \notin \mu^{(r)} \right\} \cup \left\{ \tilde{x}_{ijkl}^{(r)} \oplus \lambda_{ijkl}^{(r)} \theta^{(r)} / (i,j,k,l) \in \mu^{(r)} \right\}.$$

$$I^{(r+1)} = \left\{ I^{(r)} \cup \{(i_0,j_0,k_0,l_0)\} \right\} \setminus \{\text{la case associée à la variable sortante}\}$$

(d) Répéter les étapes **1)**...**4)**.

2.5 Etude numérique comparative

Dans cette section, nous allons donner quelques expérimentations numériques pour tester l'efficacité des algorithmes adaptés. Signalons que nous les avons implémenté sur un intel(R) core (TM) I3 dans l'environnement Microsoft Windows. Ces algorithmes sont totalement écrits en Matlab.

Nous désignons par :

- $\tilde{x}_B^{(r)}$: L'ensemble des composantes de base associé à la solution de base à l'itération r.
- $\tilde{x}_H^{(r)}$: L'ensemble des composantes hors base à l'itération r.
- $M^- N$: La taille du problème, où $M = m+n+p+q$ et $N = mnpq$.
- Z^* : La valeur optimale.
- $\Re$: La fonction de classement d'un nombre flou.
- Itr : Le nombre d'itérations.
- T : Le temps d'exécution en millisecondes.

2.5.1 Exemple illustratif d'un problème de transport flou à quatre indices de type 1

Soit un problème de transport flou à quatre indices de type 1 avec $m = n = p = q = 2$ dont les données sont présentées dans les tableaux suivants :

Puisque on a :

$$\sum_{i=1}^{2} \tilde{\alpha}_i =_{\Re} \sum_{j=1}^{2} \tilde{\beta}_j =_{\Re} \sum_{l=1}^{2} \gamma_k =_{\Re} \sum_{l=1}^{2} \tilde{\delta}_l = 11$$

Donc, ce problème de transport flou à quatre indices de type 1 admet une solution réalisable.

$\tilde{\alpha}_1$	$\tilde{\alpha}_2$	$\tilde{\beta}_1$	$\tilde{\beta}_2$	$\tilde{\gamma}_1$	$\tilde{\gamma}_2$	$\tilde{\delta}_1$	$\tilde{\delta}_2$
$(2,4,7,8)$ $\Re$=5.25	$(3,5,6,9)$ $\Re$=5.74	$(1,2,5,7)$ $\Re$=3.75	$(4,7,8,10)$ $\Re$=7.25	$(2,4,6;8)$ $\Re$=5	$(3,5,7,9)$ $\Re$=6	$(0,4,6,9)$ $\Re$=4.75	$(5,5,7,8)$ $\Re$=6.25

TABLE 2.1 – Tableau des quantités de $\tilde{\alpha}_i, \tilde{\beta}_j, \tilde{\gamma}_k$ et $\tilde{\delta}_l$

c_{1111}	c_{1112}	c_{1121}	c_{1122}	c_{1211}	c_{1212}	c_{1221}	c_{1222}
1	6	5	6	8	10	1	2
c_{2111}	c_{2112}	c_{2121}	c_{2122}	c_{2211}	c_{2212}	c_{2221}	c_{2222}
9	6	8	8	1	3	4	4

TABLE 2.2 – Matrice des coûts

Application de l'algorithme FRAM4

- La solution de base réalisable initiale obtenue en utilisant l'algorithme FRAM4 est $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$\tilde{x}_B^{(0)} = \Big\{\tilde{x}_{2211}^{(0)} = (0,4,6,9), \tilde{x}_{1222}^{(0)} = (-5,1,4,10), \tilde{x}_{1122}^{(0)} = (-8,0,6,13), \\ \tilde{x}_{2112}^{(0)} = (-7,-2,2,8), \tilde{x}_{2122}^{(0)} = (-14,-3,4,16)\Big\}.$$

  La valeur de l'objectif associée à $\tilde{x}^{(0)}$ est :

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} c_{ijkl}\tilde{x}_{ijkl}^{(0)} = (-212,-30,94,283)$$

  Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 33.75$.
- Le test d'optimalité en phase 2 montre que $\tilde{x}^{(0)}$ n'est pas optimale. Nous avons besoin de 3 itérations pour obtenir l'optimum où

$$\tilde{x}_B^{(3)} = \Big\{\tilde{x}_{2211}^{(3)} = (-28,-6,9,29), \tilde{x}_{1222}^{(3)} = (-14,-2,6,16), \tilde{x}_{1111}^{(3)} = \\ (-29,-5,12,37), \tilde{x}_{2212}^{(3)} = (-7,-2,2,8), \tilde{x}_{2222}^{(3)} = (-43,-8,16,53)\Big\}.$$

  La valeur optimale de la fonction objectif est

$$\tilde{Z}^{(3)} = \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} c_{ijkl}\tilde{x}_{ijkl}^{(3)} = (-161,-22,72,217)$$

  Sa fonction de classement est $\Re(\tilde{Z}^{(3)}) = 26.5$.

Application de l'algorithme FVAM4

- La solution de base réalisable initiale obtenue en utilisant l'algorithme FVAM4 est donnée par $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$\tilde{x}_B^{(0)} = \Big\{\tilde{x}_{1111}^{(0)} = (1,2,5,7), \tilde{x}_{2211}^{(0)} = (-7,-1,4,8), \tilde{x}_{1222}^{(0)} = (-5,-1,5,7), \\ \tilde{x}_{2222}^{(0)} = (-4,2,6,14), \tilde{x}_{2212}^{(0)} = (-19,-5,5,20)\Big\}.$$

  La valeur de l'objectif associée à $\tilde{x}^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} c_{ijkl}\tilde{x}_{ijkl}^{(0)} = (-212,-30,94,283)$$

  Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 26.5$.
- Le test d'optimalité en phase 2 montre que $\tilde{x}^{(0)}$ est optimale.

$M \times N$	Algorithme FRAM4				Algorithme VRAM4			
	Z^*	$\Re$	itr	T	Z^*	$\Re$	itr	T
10×36	$(-1394, 191, 967, 3765)$	882.25	6	57.566	$(-3060, -883, 1976, 5496)$	882.25	2	50.382
14×120	$(-10117, -3036, 4752, 13414)$	1253.2	6	51.69	$(-5830, -1354, 3046, 9151)$	1253.2	6	51.69
18×315	$(-4968, -1560, 2385, 6940)$	699.25	28	161.6	$(-11556, -2594, 3464.5, 13482)$	699.25	17	93.805
30×3024	$(-25716, -8393.2, 8676.4, 26797)$	341.05	77	1655.6	$(-23206, -7571, 7922.6, 24218)$	341.05	93	2023.4
37×6000	$(-2764.9, -821, 1096.4, 3163.4)$	168.48	82	3513.2	$(-17637, -5807.4, 6082.8, 18035)$	168.48	103	4063.7
40×9216	$(-23285, -9529.3, 9671, 24191)$	261.83	181	10977	$(-409230, -159890, 1.60070, 410100)$	261.83	159	9597.5
44×13440	$(-3478.4, -1229.2, 1414.3, 4079.8)$	196.63	150	11831	$(-201380, -79936, 80109, 201990)$	196.63	172	13929
50×21600	$(-4516.6, -1214.5, 1460.2, 4970)$	174.78	170	20395	$(-144480, -32758, 33001, 144930)$	174.78	189	23095
63×59976	$(-2836, -851, 1117, 3150)$	145	178	408.75	$(-1.8714e^6 -6.6346e^5 6.6372e^5, 1.8718e^6)$	145	311	$1.0227e^5$
68×79800	$(-4004, -1021, 1288, 4337)$	150	327	$1.5135e^5$	$(-2.0815e^6, -5.4153e^5, 5.418e^5, 2.0818e^6)$	150	428	$1.7775e^5$

TABLE 2.3 – FRAM4 et FVAM4 en résolvant PTF4 de type 1

Commentaires

— A partir de ce tableau, nous pouvons conclure que les algorithmes adaptés peuvent résoudre différents PTF4 de type 1 avec différentes tailles, ce qui montre leur efficacité.

— Pour les problèmes à petite taille, l'algorithme adapté basé sur FVAM4 atteint la solution optimale en moins d'itérations, mais pour les problèmes à grande taille, FRAM4 a tendance d'être le plus rapide.

2.5.2 Exemple illustratif d'un problème de transport flou à quatre indices de type 2

Etant donné un exemple d'un problème de transport flou à quatre indices de type 2 avec $m = 3$ et $n = p = q = 2$ dont les données sont présentées dans les tableaux suivants :

$\tilde{c}_{1111}$ $(1,2,3,4)$ $\Re$=2.5	$\tilde{c}_{1112}$ $(1,3,4,6)$ $\Re$=3.5	$\tilde{c}_{1121}$ $(-1,0,1,2)$ $\Re$=0.5	$\tilde{c}_{1122}$ $(5,7,8,11)$ $\Re$=7.75	$\tilde{c}_{1211}$ $(9,9,10,12)$ $\Re$=10	$\tilde{c}_{1212}$ $(9,11,12,14)$ $\Re$=11
$\tilde{c}_{1221}$ $(5,8,9,12)$ $\Re$=6.5	$\tilde{c}_{1222}$ $(0,1,2,3)$ $\Re$=1.5	$\tilde{c}_{2111}$ $(5,6,7,8)$ $\Re$=5.5	$\tilde{c}_{2112}$ $(3,5,6,8)$ $\Re$=8.5	$\tilde{c}_{2121}$ $(12,15,16,19)$ $\Re$=15.5	$\tilde{c}_{2122}$ $(7,9,10,12)$ $\Re$=9.5
$\tilde{c}_{2211}$ $(5,6,7,8)$ $\Re$=6.5	$\tilde{c}_{2212}$ $(1,6,7,12)$ $\Re$=6.5	$\tilde{c}_{2221}$ $(5,10,12,17)$ $\Re$=11	$\tilde{c}_{2222}$ $(5,7,8,10)$ $\Re$=7.5	$\tilde{c}_{3111}$ $(11,14,16,19)$ $\Re$=15	$\tilde{c}_{3112}$ $(12,15,16,17)$ $\Re$=15
$\tilde{c}_{3121}$ $(7,10,10,15)$ $\Re$=10.5	$\tilde{c}_{3122}$ $(1,2,3,4)$ $\Re$=2.5	$\tilde{c}_{3211}$ $(14,15,16,19)$ $\Re$=16	$\tilde{c}_{3212}$ $(12,15,16,19)$ $\Re$=15.5	$\tilde{c}_{3221}$ $(10,12,15,17)$ $\Re$=13.8	$\tilde{c}_{3222}$ $(5,8,9,12)$ $\Re$=8.5

TABLE 2.4 – Matrice des coûts

α_1	α_2	α_3	β_1	β_2	γ_1	γ_2	δ_1	δ_2
10	5	10	8	17	10	15	12	13

TABLE 2.5 – Tableau des quantités $\alpha_i, \beta_j, \gamma_k$ et δ_l .

Puisque on a :

$$\sum_{i=1}^{2} \alpha_i =_{\Re} \sum_{j=1}^{2} \beta_j =_{\Re} \sum_{l=1}^{2} \gamma_k =_{\Re} \sum_{l=1}^{2} \delta_l = 25$$

Ainsi, ce problème de transport flou à quatre indices de type 2 admet une solution réalisable.

Application de l'algorithme FRAM4

- La solution de base réalisable initiale obtenue en utilisant l'algorithme FRAM4 est donnée par $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$x_B^{(0)} = \left\{ x_{3122}^{(0)} = 8, x_{1222}^{(0)} = 5, x_{2211}^{(0)} = 5, x_{1211}^{(0)} = 3, x_{3211}^{(0)} = 2 \right\}.$$

La valeur de l'objectif associée à $x^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl}x_{ijkl}^{(0)} = (98, 120, 145, 177)$$

Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 135$.

- Le test d'optimalité en phase 2 montre que $x^{(0)}$ n'est pas optimale. On a besoin de 3 itérations pour obtenir l'optimum où

$$x_B^{(3)} = \left\{x_{3122}^{(0)} = 3, x_{1222}^{(0)} = 5, x_{2211}^{(0)} = 5, x_{3221}^{(0)} = 2, x_{3111}^{(0)} = 5, x_{3222}^{(0)} = 5\right\}.$$

La valeur optimale de la fonction objective est

$$Z^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} c_{ijkl}x_{ijkl}^{(3)} = (78, 115, 44, 181)$$

Sa fonction de classement est $\Re(Z^{(3)}) = 129.5$.

Application de l'algorithme FVAM4

- La solution de base réalisable initiale obtenue en utilisant l'algorithme FVAM4 est donnée par $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$x_B^{(0)} = \left\{x_{3122}^{(0)} = 8, x_{1222}^{(0)} = 5, x_{2211}^{(0)} = 5, x_{1221}^{(0)} = 2, x_{1211}^{(0)} = 3, x_{3211}^{(0)} = 2\right\}.$$

La valeur de l'objectif associée à $x^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl}x_{ijkl}^{(0)} = (98, 120, 145, 177)$$

Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 135$.

- Le test d'optimalité en phase 2 montre que $x^{(0)}$ n'est pas optimale. On a besoin de 3 itérations pour obtenir l'optimum où

$$x_B^{(3)} = \left\{x_{3122}^{(3)} = 3, x_{1222}^{(3)} = 5, x_{2211}^{(3)} = 5, x_{3221}^{(3)} = 2, x_{1111}^{(3)} = 5, x_{3222}^{(3)} = 5\right\}$$

La valeur optimale de la fonction objectif est

$$\tilde{Z}^{(3)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl}x_{ijkl}^{(3)} = (78, 115, 44, 181)$$

Sa fonction de classement est $\Re(\tilde{Z}^{(3)}) = 129.5$.

$M \times N$	Algorithme FRAM4				Algorithme VRAM4			
	Z^*	$\Re$	itr	T	Z^*	$\Re$	itr	T
10×36	(588, 1582, 2325, 3239)	1933.5	1	36.799	(588, 1582, 2325, 3239)	1933.5	1	31.682
13×90	(1049, 1941, 2739, 3836)	2391.3	4	46.392	(1049, 1941, 2739, 3836)	2391.3	5	53.915
17×288	(457, 1236.5, 2498.5, 4133)	2081.2	12	78.013	(457, 1236.5, 2498.5, 4133)	2081.2	17	88.795
25×1400	(443, 917, 1602, 3474)	1609	31	487.05	(443, 917, 1602, 3474)	1609	32	440.92
29×2268	(823, 1066, 2057, 2473)	1604.8	45	864.6	(823, 1066, 2057, 2473)	1604.8	35	674.52
33×4320	(453, 856.73, 1446, 2689.4)	1361.3	141	10088	(453, 856.73, 1446, 2689.4)	1361.3	120	10975
54×32760	(438, 986.25, 1838, 2880.2)	1535.6	338	68943	(438, 986.25, 1838, 2880.2)	1535.6	278	56432
57×40500	(308.26, 836, 1313.6, 2006)	1116	282	57638	(308.26, 836, 1313.6, 2006)	1116	241	48984
62×56448	(426.44, 785.78, 1336.7, 2351.8)	1225.2	393	1078450	(425.93, 785.11, 1337.7, 2351.9)	1225.2	364	99338
68×79800	(340.91, 783.36, 1310.1, 2479.8)	1228.5	460	200220	(340.91, 783.36, 1310.1, 2479.8)	1228.5	439	188780

TABLE 2.6 – FRAM4, et FVAM4 en résolvant PTF4 de type 2

Commentaires

— A partir de ce tableau, nous pouvons conclure que les deux algorithmes appliqués peuvent résoudre différents PTF4 de type 2 avec des tailles différentes, ce qui montre leur efficacité.

— Pour les PTF4 de type 2 à petite taille, l'algorithme adapté FRAM4 atteint la solution optimale en moins d'itérations, mais pour les problèmes à grande taille, FVAM4 a tendance d'être le plus rapide.

2.5.3 Exemple illustratif d'un problème de transport flou à quatre indices de type 3

Considérons un problème de transport à quatre indices de type 3 (totalement flou) avec $(m = n = p = q = 2)$, les quantités de $\tilde{\alpha}_i, \tilde{\beta}_j, \tilde{\gamma}_k, \tilde{\delta}_l$, et $\tilde{c}_{ijkl}$ sont donnés par les tableaux suivants :

$\tilde{c}_{1111}$	$\tilde{c}_{1112}$	$\tilde{c}_{1121}$	$\tilde{c}_{1122}$	$\tilde{c}_{1211}$	$\tilde{c}_{1212}$	$\tilde{c}_{1221}$	$\tilde{c}_{1222}$
$(4,5,6)$ $\Re = 5$	$(0,2,7)$ $\Re = 2.75$	$(1,3,5)$ $\Re = 3$	$(5,6,9)$ $\Re = 6.5$	$(3,5,6)$ $\Re = 4.75$	$(4,6,9)$ $\Re = 6.25$	$(6,7,9)$ $\Re = 7.25$	$(2,4,5)$ $\Re = 3.75$
$\tilde{c}_{2111}$	$\tilde{c}_{2112}$	$\tilde{c}_{2121}$	$\tilde{c}_{2122}$	$\tilde{c}_{2211}$	$\tilde{c}_{2212}$	$\tilde{c}_{2221}$	$\tilde{c}_{2222}$
$(5,6,8)$ $\Re = 6.25$	$(6,8,10)$ $\Re = 8$	$(2,3,7)$ $\Re = 3.75$	$(6,8,12)$ $\Re = 8.5$	$(7,9,11)$ $\Re = 9$	$(3,9,7)$ $\Re = 7$	$(3,4,5)$ $\Re = 4$	$(4,6,10)$ $\Re = 6.5$

TABLE 2.7 – Matrice des coûts

$\tilde{\alpha}_1$	$\tilde{\alpha}_2$	$\tilde{\beta}_1$	$\tilde{\beta}_2$	$\tilde{\gamma}_1$	$\tilde{\gamma}_2$	$\tilde{\delta}_1$	$\tilde{\delta}_2$
$(3,7,7)$ $\Re = 6$	$(1,2,7)$ $\Re = 3$	$(3,4,8)$ $\Re = 4.75$	$(1,5,6)$ $\Re = 4.25$	$(2,2,3)$ $\Re = 2.25$	$(2,7,11)$ $\Re = 6.75$	$(0,4,6)$ $\Re = 3.5$	$(4,5,8)$ $\Re = 5.5$

TABLE 2.8 – Tableau des quantités de $\tilde{\alpha}_i, \tilde{\beta}_j, \tilde{\gamma}_k$ et $\tilde{\delta}_l$

Ce PTF4 admet une solution réalisable car :

$$\sum_{i=1}^{2} \tilde{\alpha}_i =_{\Re} \sum_{j=1}^{2} \tilde{\beta}_j =_{\Re} \sum_{l=1}^{2} \gamma_k =_{\Re} \sum_{l=1}^{2} \tilde{\delta}_l = (4,9,14)$$

On a $M = 8$ et $N = 16$.

Application de l'algorithme FLC4

- Trouver la solution réalisable initiale
  - Prendre $E_b = \emptyset$,
  - $\min \tilde{c}_{ijkjl} = \tilde{c}_{1112}$,
  - Déterminer $\tilde{x}_{1112}$:

$$\begin{aligned} \tilde{x}_{1112} &= \min\left(\tilde{\alpha}_1, \tilde{\beta}_1, \tilde{\gamma}_1, \tilde{\delta}_2\right) \\ &= \min\left((3,7,7)_{\Re=6}, (3,4,8)_{\Re=4.75}, (2,2,3)_{\Re=2.25}, (4,5,8)_{\Re=5.5}\right) \\ &= (2,2,3)_{\Re=2.25}. \end{aligned}$$

  - Ajouter $(1,1,1,2)$ à E_b,
  - Actualiser $\tilde{\alpha}_1, \tilde{\beta}_1, \tilde{\gamma}_1$ et $\tilde{\delta}_2$ comme suit :

$$\tilde{\alpha}_1 = (0,5,5),$$

$$\tilde{\beta}_1 = (0,2,6),$$

$$\tilde{\gamma}_1 = (-1,0,1),$$

$$\tilde{\delta}_2 = (1,3,6).$$

— Pour tout $(i, j, l) \neq (1, 1, 2)$, prendre $\tilde{x}_{ij1l}$ égal à zéro flou et saturer $\tilde{c}_{ij1l}$.
— Répéter jusqu'à ce que toutes les variables $\tilde{x}_{ijkl}$ soient déterminées.

La solution de base réalisable initiale obtenue par FLC4 est donnée par : $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$\tilde{x}_B^{(0)} = \Big\{\tilde{x}_{1112}^{(0)} = (2,2,3), \tilde{x}_{1121}^{(0)} = (0,2,6), \tilde{x}_{1222}^{(0)} = (-6,3,5), \\ \tilde{x}_{2221}^{(0)} = (-6,2,6), \tilde{x}_{2222} = (-4,0,12)\Big\}.$$

La valeur de l'objectif associée à $\tilde{x}^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl} \otimes \tilde{x}_{ijkl}^{(0)} = (16, 33, 59.5).$$

Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 35.3750$

- Test de la dégénérescence
  Le nombre des éléments de $\tilde{x}_B^{(0)}$ est égal à $5 = M - 3$; i.e., la solution est non dégénérée.

- Phase 2
  Le test d'optimalité en phase 2 montre que cette solution n'est pas optimale. Ainsi, nous pouvons l'améliorer. $\tilde{x}^{(1)} = \tilde{x}_B^{(1)} \cup \tilde{x}_H^{(1)}$ où

$$\tilde{x}_B^{(1)} = \Big\{\tilde{x}_{1112}^{(1)} = (2,2,3), \tilde{x}_{1121}^{(1)} = (-12,2,10), \tilde{x}_{1222}^{(1)} = (-10,3,17), \\ \tilde{x}_{2121}^{(1)} = (-4,0,12), \tilde{x}_{2221}^{(1)} = (-6,2,6)\Big\}$$

  Le test d'optimalité montre que $\tilde{x}^{(1)}$ est optimale. La valeur de l'objectif associée à $\tilde{x}^{(1)}$ est

$$\tilde{Z}^{(1)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl} \otimes \tilde{x}_{ijkl}^{(1)} = (14, 29, 53.5).$$

  Sa fonction de classement est $\Re(\tilde{Z}^{(1)}) = 31.3750$

Application de l'algorithme FRAM4

- Trouver la solution réalisable initiale
  - — Prendre $E_b = \emptyset$,
  - — Calculer les coûts réduits :

$\tilde{c}^{(i=1)}$	$\tilde{c}^{(i=2)}$	$\tilde{c}^{(j=1)}$	$\tilde{c}^{(j=2)}$	$\tilde{c}^{(k=1)}$	$\tilde{c}^{(k=2)}$	$\tilde{c}^{(l=1)}$	$\tilde{c}^{(l=2)}$
$(6,7,9)$	$(7,9,11)$	$(6,8,12)$	$(7,9,11)$	$(7,9,11)$	$(6,8,12)$	$(7,9,11)$	$(6,8,12)$

TABLE 2.9 – Coûts réduits en première itération

  - — Calculer la matrice des coûts réduits :

c^*_{1111}	c^*_{1112}	c^*_{1121}	c^*_{1122}
$(-29,-28,-20)$	$(-44,-30,-18)$	$(-43,-29,-20)$	$(-40,-25,-15)$
c^*_{1211}	c^*_{1212}	c^*_{1221}	c^*_{1222}
$(-39,-29,-21)$	$(-39,-27,-17)$	$(-37,-26,-17)$	$(-42,-28,-20)$
c^*_{2111}	c^*_{2112}	c^*_{2121}	c^*_{2122}
$(-40,-29,-19)$	$(-40,-26,-16)$	$(-44,-31,-19)$	$(-41,-25,-13)$
c^*_{2211}	c^*_{2212}	c^*_{2221}	c^*_{2222}
$(-37,-27,-17)$	$(-42,-26,-20)$	$(-42,-31,-22)$	$(-44,-28,-16)$

TABLE 2.10 – Matrice des coûts réduits en première itération

  - — $\min\limits_{i,j,k,l} c^*_{ijkl} = c^*_{2221}$,
  - — Déterminer $\tilde{x}_{2221}$:

$$\begin{aligned}\tilde{x}_{2221} &= \min\left(\tilde{\alpha}_2, \tilde{\beta}_2, \tilde{\gamma}_2, \tilde{\delta}_1\right)\\ &= \min\left((1,2,7)_{\Re=3}, (1,5,6)_{\Re=4.25}, (2,7,11)_{\Re=6.75}, (0,4,6)_{\Re=3.5}\right)\\ &= (1,2,7)_{\Re=3}.\end{aligned}$$

  - — Ajouter $(2,2,2,1)$ à E_b,
  - — Actualiser $\tilde{\alpha}_2, \tilde{\beta}_2, \tilde{\gamma}_2$ et $\tilde{\delta}_1$ comme suit :

$$\tilde{\alpha}_2 = (-6,0,6),$$

$$\tilde{\beta}_2 = (-6,3,5),$$

$$\tilde{\gamma}_2 = (-5,5,10),$$

$$\tilde{\delta}_1 = (-7,2,5).$$

  - — Pour tout $(i,j,l) \neq (2,2,1)$, prendre $\tilde{x}_{2jkl}$ égal à zéro flou et saturer $\tilde{c}_{2jkl}$.
  - — Répéter jusqu'à ce que toutes les variables $\tilde{x}_{ijkl}$ soient déterminées.

La solution de base réalisable initiale obtenue par FRAM4 est donnée par : $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$ où

$$\tilde{x}_B^{(0)} = \Big\{\tilde{x}_{2221}^{(0)} = (1,2,7), \tilde{x}_{1121}^{(0)} = (-7,2,5), \tilde{x}_{1112}^{(0)} = (2,2,3), \\ \tilde{x}_{1222}^{(0)} = (-6,3,5), \tilde{x}_{1122}^{(0)} = (-10,0,18)\Big\}.$$

La valeur de l'objectif associée à $\tilde{x}^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl} \otimes \tilde{x}_{ijkl}^{(0)} = (14, 29, 53.5).$$

Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 31.3750$

- Test de la dégénérescence
  Le nombre des éléments de $\tilde{x}_B^{(0)}$ est égal à $5 = M-3$; i.e., la solution est non dégénérée.
- Phase 2
  Le test d'optimalité en phase 2 montre que cette solution est optimale.

Application de l'algorithme FVAM4

- Trouver la solution de base réalisable initiale
  — Prendre $E_b = \emptyset$,
  — Calculer les pénalités :

p_1^1	p_2^1	p_1^2	p_2^2	p_1^3	p_2^3	p_1^4	p_2^4
$(-6,1,5)$	$(-4,1,3)$	$(-6,1,5)$	$(-2,0,3)$	**(-4,3,6)**	$(-3,1,4)$	$(-3,0,6)$	$(-5,2,5)$
$\Re = 0.25$	$\Re = 0.25$	$\Re = 0.25$	$\Re = 0.25$	$\Re = 2$	$\Re = 0.75$	$\Re = 0.75$	$\Re = 1$

TABLE 2.11 – Pénalités dans la première itération

- $\max(p_i^1, p_j^2, p_k^3, p_l^4) = p_1^3$,
- $\min \tilde{c}_{ij1l} = \tilde{c}_{1112}$,

— Déterminer $\tilde{x}_{1112}$:

$$\begin{aligned} \tilde{x}_{1112} &= \min\left(\tilde{\alpha}_1, \tilde{\beta}_1, \tilde{\gamma}_1, \tilde{\delta}_2\right) \\ &= \min\left((3,7,7)_{\Re=6}, (3,4,8)_{\Re=4.75}, (2,2,3)_{\Re=2.25}, (4,5,8)_{\Re=5.5}\right) \\ &= (2,2,3)_{\Re=2.25}. \end{aligned}$$

— Ajouter $(1,1,1,2)$ à E_b,

— Actualiser $\tilde{\alpha}_1, \tilde{\beta}_1, \tilde{\gamma}_1$ et $\tilde{\delta}_2$:

$$\tilde{\alpha}_1 = (0, 5, 5),$$

$$\tilde{\beta}_1 = (0, 2, 6),$$

$$\tilde{\gamma}_1 = (-1, 0, 1),$$

$$\tilde{\delta}_2 = (1, 3, 6).$$

— Pour tout $(i, j, l) \neq (1, 1, 2)$, soit $\tilde{x}_{ij1l}$ égal à zéro flou et saturer $\tilde{c}_{ij1l}$.
— Répéter jusqu'à ce que toutes les variables $\tilde{x}_{ijkl}$ soient déterminées.

La solution de base réalisable initiale obtenue par FVAM4 est donnée par : $\tilde{x}^{(0)} = \tilde{x}_B^{(0)} \cup \tilde{x}_H^{(0)}$, où

$$\tilde{x}_B^{(0)} = \Big\{\tilde{x}_{1112}^{(0)} = (2, 2, 3), \tilde{x}_{1222}^{(0)} = (1, 3, 6), \tilde{x}_{1121}^{(0)} = (-6, 2, 4), \tilde{x}_{2221}^{(0)} = (-5, 2, 5), \tilde{x}_{2121}^{(0)} = (-4, 0, 12)\Big\}.$$

La valeur de l'objectif associée à $\tilde{x}^{(0)}$ est

$$\tilde{Z}^{(0)} =_{\Re} \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}\sum_{l=1}^{2} \tilde{c}_{ijkl} \otimes \tilde{x}_{ijkl}^{(0)} = (14, 29, 53.5)$$

Sa fonction de classement est $\Re(\tilde{Z}^{(0)}) = 31.3750$

- Test de la dégénérescence
  Le nombre de éléments de $\tilde{x}_B^{(0)}$ est égal à $5 = M - 3$; donc la solution est non dégénérée.
- Phase 2
  Le test d'optimalité en phase 2 montre que $\tilde{x}^{(0)}$ est optimale et n'a pas besoin d'amélioration.

Dimension ($M \times N$)			12×81	16×265	20×625	24×1296	32×4096	40×1000	64×65536
phase 1	$\tilde{Z}^{(0)}$	FLC4	(-2998.8 2093.8 7131.8)	(-21125 1653.3 24298)	(-54986 1052.3 57223)	(-75878 2136.3 79614)	(-62829 1166.5 65238)	(-33481 348.5 34377)	(-77002 1750 80003)
		FRAM4	(-1948.8 2043.8 6307)	(-23479 1622.5 26517)	(-70631 931 73088)	(-45375 1895.8 49255)	(-40825 1021.3 43588)	(-61920 314.75 63186)	(-1.4375e5 1432 1.4748e5)
		FVAM4	(-2836.3 2103.3 6971.3)	(-29170 1440.5 32234)	(-25078 1094.8 26875)	(-43860 1865.8 47852)	(-60253 1192.3 62500)	(-84944 360 85863)	(-1.1649e5 1417.3 1.199e5)
	$\Re(\tilde{Z}^{(0)})$	FLC4	2080.1	1619.9	1085.6	2002.3	1185.5	398.25	1625.3
		FRAM4	2111.4	1570.8	1079.8	1917.7	1201.4	474	1649.1
		FVAM4	2085.4	1486.4	996.69	1931	1158	409.69	1561
	T(ms)	FLC4	1.5356	2.2069	4.0085	12.481	13.846	15.576	17.398
		FRAM4	3.0169	6.7252	9.512	31.602	50.462	96.994	701.71
		FVAM4	3.6408	6.794	7.9264	30.963	30.922	45.034	168.96
phase 2	$\tilde{Z}^{*}$	FLC4	(-2818 2015.8 6833.8)	(-12709 1343.3 15325)	(-27951 864.25 29806)	(-52205 1455.3 54931)	(-28621 847.66 30335)	(-18536 284.85 19198)	(-43864 682.97 45460)
		FRAM4	(-1900.8 2015.7 5916.5)	(-10040 1347.3 12649)	(-32677 874.75 34511)	(-28878 1455.3 31605)	(-19155 847.66 20869)	(-41649 295.58 42290)	(-72237 704.61 73789)
		FVAM4	(-2775.2 2015.8 6791)	(-17652 1345.6 20264)	(-17330 868.75 19176)	(-29654 1455.3 32381)	(-37737 848.3 39450)	(-31037 290.22 31689)	(-79578 684.73 81170)
	$\Re(\tilde{Z}^{*})$	FLC4	2011.8	1325.9	895.87	1409.3	852.24	308.01	740.36
		FRAM4	2011.8	1325.9	895.87	1409.3	852.24	308.0	740.36
		FVAM4	2011.8	1325.9	895.87	1409.3	852.24	308.01	740.36
	iter	FLC4	4	12	14	50	52	69	345
		FRAM4	3	14	16	48	70	68	418
		FVAM4	2	8	12	48	57	68	354
	T(ms)	FLC4	8.8915	50.207	145.4	1507.8	8845.3	56340	1.0578e5
		FRAM4	7.1789	58.461	162.94	1347.3	11605	58289	1.2892e5
		FVAM4	5.0874	34.99	125.01	1318.6	9576.5	56062	1.0562e5
Temps total (ms)		FLC4	10.427	52.414	149.41	1520.3	8859.1	56356	1.058e5
		FRAM4	10.196	65.186	172.45	1378.9	11655	58386	1.2962e5
		FVAM4	8.7282	41.784	132.94	1349.6	9607.4	56107	1.0579e5

TABLE 2.12 – Efficacité des algorithmes FLC4, FRAM4, et FVAM4

Dimension ($M \times N$)			30×3136	35×5760	37×7200	41×10800	80×16000	113×630000
phase 1	$\tilde{Z}^{(0)}$	FLC4	(-1.4195e5 1507.8 1.4447e5)	(-30855 2086.8 34891)	(-57572 1157.3 59759)	(-1.2796e5 1602 1.3161e5)	(-2.6674e5 764.25 2.7078e5)	(-5.4236e5 854 5.4413e5)
		FRAM4	(-15678 1711.3 18334)	(-26876 1920 30676)	(-39550 1127.3 41836)	(-57375 1496.3 60930)	(-6.2504e5 1357 6.2814e5)	(-1.9974e5 920.25 2.0114e5)
		FVAM4	(-30597 1508.5 33267)	(-49972 1777.8 52403)	(-38357 1067.8 40487)	(-47532 1353.8 51708)	(-1.5441e5 1256 1.5731e5)	(-234661 885.25 236372)
	$\Re(\tilde{Z}^{(0)})$	FLC4	1384.1	2052.3	1125.4	1714.4	1393.4	871.31
		FRAM4	1519.4	1910.1	1135.2	1636.8	1454.9	808.81
		FVAM4	1421.9	1496.6	1066.4	1721	1352.8	870.38
	T(ms)	FLC4	13.73	6.0158	15.905	16.345	63.173	211.78
		FRAM4	43.844	62.404	75.093	101.12	2148.7	15467
		FVAM4	30.119	24.991	39.1	72.504	382.51	2137
phase 2	$\tilde{Z}_*$	FLC4	(-76187 1080.6 78056)	(-16392 853.97 17984)	(-46683 800.33 48381)	(-72990 933.75 75036)	(-83399 555.63 84528)	(-2.375e5 417.71 2.3825e5)
		FRAM4	(-10714 1108.6 12528)	(-15701 853.97 17293)	(-25600 795.83 27307)	(-39633 940.25 41666)	(-2.8724e5 530.74 2.8842e5)	(-96678 412.08 97440)
		FVAM4	(-22150 1137.7 23905)	(-29421 871.09 30979)	(-34851 794 36562)	(-47532 1353.8 51708)	(-97204 540.54 98363)	(-1.4737e5 418.57 1.4812e5)
	$\Re(\tilde{Z}_*)$	FLC4	1007.6	825.09	824.65	978.38	560	396.65
		FRAM4	1007.6	825.09	824.65	978.38	560	396.65
		FVAM4	1007.6	825.09	824.65	978.38	560	396.65
	iter	FLC4	58	86	83	130	565	1225
		FRAM4	68	86	79	123	549	1095
		FVAM4	60	83	88	91	512	1074
	T(ms)	FLC4	1103.7	3452.1	3785	7341	4.3418e5	3.6585e6
		FRAM4	1959.5	2813.9	3334.3	8074.2	4.1115e5	3.1405e6
		FVAM4	1245.2	4161.7	3704.6	5824.4	3.9497e5	3.01e6
Temps total (ms)		FLC4	1117.4	3458.1	3800.9	7357.4	4.3424e5	3.6587e6
		FRAM4	2003.4	2876.3	3409.4	8175.3	4.133e5	3.156e6
		FVAM4	1275.3	4186.7	3743.7	5896.9	3.9535e5	3.0121e6

TABLE 2.13 – FLC4, FRAM4, et FVAM4 en résolvant PTF4 totalement flou

Commentaires

— Ces tableaux montrent que les algorithmes proposés peuvent résoudre efficacement le PTF4 totalement flou avec une large gamme de dimensions allant de 8×16 à plus de 113×630000.
— La fonction de classement de la solution optimale est la même, ce qui signifie que nos algorithmes sont robustes et convergent vers la solution optimale floue et ne dépendent pas de la solution initiale.
— Les résultats montrent la robustesse de notre méthode en phase 2 lors de la recherche de la solution optimale, à partir d'une solution initiale, même en cas de dégénérescence.
— La solution de base réalisable initiale obtenue par l'algorithme FLC4 est loin de l'optimum pour la plupart des PTF4, son avantage est qu'il est rapide et simple.
— La solution de base réalisable initiale obtenue par les algorithmes FVAM4 ou FRAM4 est en général, très proche de la solution optimale.
— Dans la plupart des cas, la méthode FVAM4 nécessite moins de temps et un plus petit nombre d'itérations par rapport aux méthodes FRAM4 et FLC4. Par conséquent, FVAM4 est préférable à utiliser pour résoudre les problèmes de transport totalement flous à quatre indices de grande taille.

CHAPITRE 3

MÉTAHEURISTIQUES POUR RÉSOUDRE UN PROBLÈME DE TRANSPORT FLOU À QUATRE INDICES

3.1 Introduction

L'un des principes les plus fondamentaux de notre monde, qui occupe une place importante dans le monde informatique, industriel, etc., est la recherche d'un état optimal. En effet ; de nombreux problèmes scientifiques, sociaux, économiques et techniques ont des paramètres qui peuvent être ajustés pour produire un résultat plus souhaitable. Ceux-ci peuvent être considérés comme des problèmes d'optimisation et leur résolution est un sujet central en recherche opérationnelle. Plusieurs méthodes ont été conçues pour résoudre ces problèmes.

Ces méthodes se classent en deux catégories bien distinctes. D'une part, les méthodes exactes, cherchant à trouver de manière certaine la solution optimale en considérant l'ensemble des solutions possibles. D'autre part, les méthodes approchées, qui se contentent de rechercher une solution "de bonne qualité". Notamment les problèmes difficiles, par exemple ceux qui présentent de nombreux minimums locaux, en déterminant des solutions qui ne sont pas rigoureusement optimales, mais qui s'en approchent. Ces méthodes, appelées heuristiques et métaheuristiques, ils se basent généralement sur des phénomènes physiques, biologiques, sociopsychologiques, et souvent faire appel au hasard.

Contrairement aux méthodes exactes qui garantissent des solutions exactes, Une heuristique est un algorithme qui a pour objectif de trouver une solution réalisable sans garantie d'optimalité. On fait appel aux métaheuristiques pour résoudre les problèmes difficiles lorsque la résolution en utilisant les méthodes exactes est de complexité exponentielle.

3.2 Caractéristiques des métaheuristiques

On peut définir deux aspects communs dans les algorithmes d'optimisation basés sur la population : exploration et exploitation. L'exploration est la capacité d'élargir l'espace de recherche, alors que l'exploitation est la capacité de trouver l'optimum autour d'une bonne solu-

tion. Dans ses premières itérations, un algorithme de recherche heuristique explore l'espace de recherche pour trouver de nouvelles solutions tout en évitant le piégeage dans un optimum local. Pour faire une recherche de haute performance, il faut faire un compromis approprié entre exploration et exploitation. Lorsque l'évolution de la méthode de résolution est prévisible et ne laisse aucune place au hasard, celle-ci est qualifiée d'une méthode déterministe. On s'intéresse plus particulièrement aux métaheuristiques qui sont des méthodes stochastiques. Le mot métaheuristique vient de la composition de deux mots grecs :

— Heuristique qui vient du verbe heuriskein (euriskein) et qui signifie trouver ;
— Méta qui est un suffixe signifiant au-delà, dans un niveau supérieur.

3.3 Avantages et inconvénients des méthodes métaheuristiques

Parmi les avantages et les inconvénients des méthodes métaheuristiques on peut citer :

Avantages

— Grâce à leur caractère stochastique, il s'agit de méthodes d'optimisation globale.
— Grâce à l'utilisation d'une population, il s'agit de méthodes facilement parallélisables.
— Ces méthodes sont également très stables par rapport aux erreurs numériques commises sur la fonction objectif.
— La vitesse de convergence est très rapide en comparant à une méthode déterministe pour la résolution des problèmes de grandes tailles.

Inconvénients

— Leur coût de calcul important lié au grand nombre d'évaluations de la fonction objectif.
— Le choix des paramètres est très influent sur la qualité des résultats obtenus.
— Ils ne respectent pas les bornes de domaine de recherche.

Dans ce qui suit, on présente quelques fondations nécessaires ainsi que les descriptions concernant les deux méthodes métaheuristiques qu'on va les utilisés pour résoudre le problème de transport flou à quatre indices.

3.4 Algorithmes génétiques (AG)

Les algorithmes génétiques (AG) sont des algorithmes d'optimisation stochastique fondés sur les mécanismes de la sélection naturelle et de la génétique. Ils ont été initialement développés par John Holland (1975) qui a développé les principes fondamentaux des AG dans le

cadre de l'optimisation mathématique. C'est une technique de recherche utilisant la génétique comme un modèle de résolution pour trouver des solutions approximatives aux problèmes d'optimisation.

3.4.1 Terminologie

Dans cette section, on présente quelques définitions de vocabulaire nécessaires pour une bonne compréhension.

Chromosome : Un chromosome est constitué d'une séquence finie des gènes qui peuvent prendre des valeurs appelées allèles qui sont prises dans un alphabet qui doit être judicieusement choisi pour convenir du problème étudié.

Individu : Un individu représente une solution réalisable du problème. Par analogie avec la génétique, il est codé par des chromosomes.

Codage : L'étape clé dans un algorithme génétique est comment définir et coder convenablement les variables du problème donné. La littérature définit deux types de codage : binaire et réel.

Population initiale : Plusieurs mécanismes de génération de la population initiale sont utilisés dans la littérature. Le choix de l'initialisation se fera en fonction des connaissances que l'utilisateur a sur le problème. S'il n'a pas d'informations particulières, alors une initialisation aléatoire, la plus uniforme possible afin de favoriser une exploration de l'espace de recherche maximum, sera la plus adaptée. Par ailleurs, cette étape présente un problème principal qui est celui de choix de la taille de la population. En effet, une population trop grande augmente le temps de calcul et demande un espace mémoire considérable, alors qu'une population petite conduit à l'obtention d'un optimum local.

Sélection : La sélection est une méthode qui choisit d'une manière organisée des chromosomes à partir de la population en fonction des valeurs de la fonction d'objectif. Ce procédé permet de donner aux meilleures individus, une probabilité de contribuer à la génération suivante. Cet opérateur est bien entendu une version artificielle de la sélection naturelle qui est la survie darwinienne des individus les plus adaptées.

Il existe de nombreuses techniques de sélection, parmi les plus courantes on cite : La sélection par roulette, par rang et par tournoi, . . . , etc.

Parent : Un parent est un individu de la génération courante choisi par un opérateur de sélection pour générer des nouveaux individus (dits enfants).

Enfant : Dès que deux parents sont choisis dans une génération courante, les opérateurs de croisement et de mutation vont opérer afin d'obtenir des nouveaux individus appelés "enfants".

Croisement : La naissance d'un nouvel individu, nécessite la prise aléatoire d'une partie des chromosomes de chacun des deux parents. Ce phénomène, issu de la nature est appelé croisement (crossover). La littérature définit plusieurs opérateurs de croisement selon le type de codage adapté et la nature du problème traité.

Mutation : C'est un processus où un changement mineur de code génétique est appliqué à un individu pour introduire de la diversité et ainsi d'éviter de tomber dans des optimums locaux.

Test d'arrêt : Le test d'arrêt joue un rôle primordiale dans le jugement de la qualité des individus. Son but est de nous assurer l'optimalité, de la solution finale obtenue par l'algorithme génétique.

Les critères d'arrêts sont de deux natures :

1. Arrêt après un nombre fixé à priori de générations. C'est la solution retenue lorsqu'une durée maximale de temps de calcul est imposée.
2. Arrêt lorsque la population cesse d'évoluer ou n'évolue plus. Ce test d'arrêt reste le plus objectif et le plus utilisé.

3.4.2 Principes de base des AG

Indépendamment de la problématique traitée, les algorithmes génétiques sont basés sur six principes :

1. Manière de codage d'une solution.
2. Génération d'une population initiale de taille fixe N_p, formée d'un ensemble fini de solutions, dite génération initiale.
3. Définition d'une fonction d'évaluation (fitness) permettant d'évaluer une solution et la comparer par rapport aux autres.
4. Mécanisme de sélection qui choisit les solutions pour un éventuel couplage.
5. Opérateurs génétiques pour générer des nouvelles solutions :
   — Opérateur de croisement : il manipule la structure des chromosomes des parents afin de produire des individus meilleurs. Cet opérateur est effectué selon une probabilité P_c.
   — Opérateur de mutation : il évite d'établir des populations uniformes incapables d'évoluer. Il consiste à modifier les valeurs des gènes de chromosomes selon une probabilité de mutation P_m.
6. Mécanisme d'insertion : il établit un compromis entre les solutions produites (progénitures) et les solutions productrices (les parents). En d'autre terme, et suite à des informations précises, décider qui doit rester et qui doit disparaître. Tout ceci, en sauvegardant à chaque génération une taille de la population N_p fixe.

3.4.3 Fonctionnement des AG

L'algorithme génétique présenté dans la figure 3.1, débute par une génération d'une population initiale de N_p individus, pour lesquels, nous calculons les valeurs de leur fonction objectif et nous sélectionnons les individus par une méthode de sélection. Les individus, sujets de croisement par l'opérateur de croisement, sont choisis selon une probabilité P_c. Leurs résultats peuvent être mutés par un opérateur de mutation avec une probabilité de mutation P_m. Les individus issus de ces opérateurs génétiques seront insérés avec une méthode d'insertion dans la nouvelle population dont nous évaluons la valeur de la fonction objectif de chacun de ses individus. Un test d'arrêt sera effectué pour vérifier la qualité des individus obtenus. Si ce test est vérifié alors l'algorithme s'arrête avec une solution optimale, sinon on réitère le processus sur la nouvelle génération.

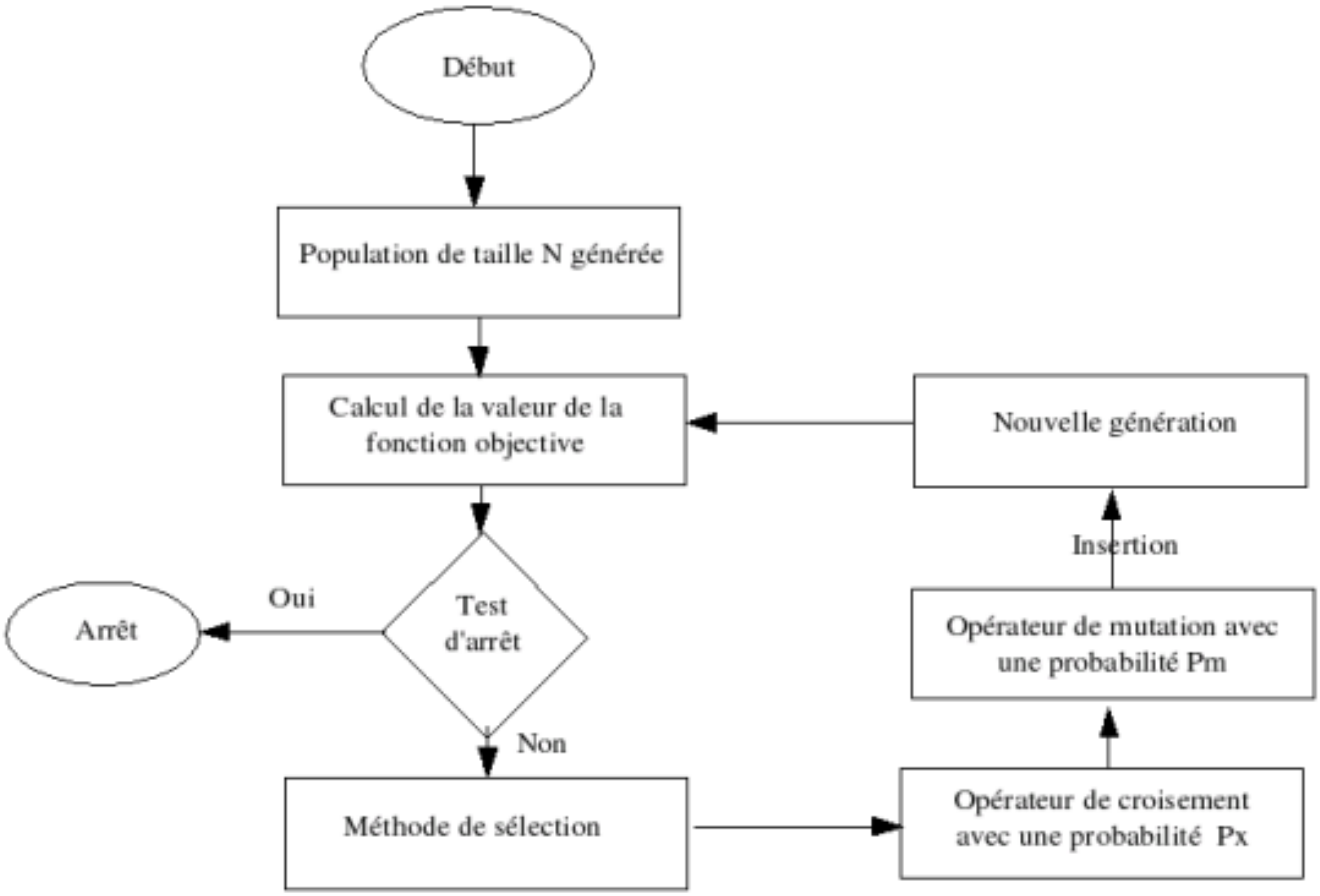

FIGURE 3.1 – L'organigramme de l'algorithme génétique

3.5 Particle Swarm Optimization (PSO)

L'optimisation par essaim particulaire (OEP), ou Particle Swarm Optimization (PSO) en anglais est un algorithme évolutionnaire qui utilise une population candidate pour développer une solution optimale au problème. Cet algorithme a été proposé par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [22]. Il s'inspire à l'origine du monde des vivants, plus précisément du comportement social des animaux évoluant en essaim, tel que les bancs de poissons et les vols groupés d'oiseaux. L'information et la mémoire de chaque individu est utilisée pour décider de son déplacement. Des règles simples, telles que "rester proche des autres individus", "aller dans la même direction" ou "aller à la même vitesse", suffisent pour maintenir la cohésion de l'essaim, et permettent la mise en uvre de comportements collectifs complexes et adaptatifs.

3.5.1 Terminologie

Particule : Une particule est l'élément essentiel de la population. Chaque particule représente une solution potentielle au problème d'optimisation.

Swarm : Dans la terminologie anglaise warm présente un essaim des animaux, tels que les nuées d'oiseaux et les bancs de poissons. Un essaim est composé d'un ensemble des particules.

Voisinage : Le voisinage d'une particule c'est l'ensemble des particules qui ont une influence directe sur son comportement et son état futur. Dans l'algorithme PSO, il y a deux types de voisinage ; le premier est partiel où un nombre limité des particules influencent sur elle. Le deuxième est complet, dans ce cas on suppose que chaque particule partage son comportement avec toutes les particules.

Position : La position d'une particule présente un codage pour l'ensemble variable modélisant le problème d'optimisation étudié.

Vitesse : La vitesse représente la direction du mouvement de la particule dans l'itération suivante. En effet, à chaque itération la particule ajuste et modifie sa position en fonction de trois types de vitesse : inertie, $Pbest_o$, $Gbest$.

Inertie : L'inertie d'une particule est son attitude à rester dans son trajectoire courante.

$Pbest_o$ **:** Chaque particule mémorise sa meilleure ancienne position. La vitesse $Pbest_o$ c'est la tendance de chaque particule d'aller vers sa meilleure ancienne position.

$Gbest$ **:** La vitesse $Gbest$ c'est la tendance de chaque particule d'aller vers la meilleure position de son voisinage.

3.5.2 Principes de base de PSO

L'essaim de particules correspond à une population d'agents simples, appelés particules. Chaque particule est considérée comme une solution du problème, où elle possède une position (vecteur solution) et une vitesse. De plus, chaque particule possède une mémoire lui permettant de se souvenir de sa meilleure performance (en position et en valeur) et de la meilleure performance atteinte par les particules "voisines". Dans un espace de recherche de dimension D, la particule o de l'essaim est modélisée à l' instant t par son vecteur position x_o^t et par son vecteur vitesse v_o^t. La qualité de sa position est déterminée par la valeur de la fonction objectif en ce point. Cette particule garde en mémoire la meilleure position par laquelle elle est déjà passée, que l'on note $pbest_o$. La meilleure position atteinte par les particules de l'essaim est notée $gbest$. Nous nous référons à la version globale de PSO, où toutes les particules de l'essaim sont considérées comme voisines de la particule o, d'où la notation :

$$\begin{cases} v_o^{t+1} = wv_o^t + c_1r_1(pbest_o - x_o^t) + c_2r_2(gbest - x_o^t) \\ x_o^{t+1} = x_o^t + v_o^{t+1} \end{cases}$$

Où v_o^t est la vitesse de particule o à l' instant t. x_o^t est la position de particule o à l'instant t. w, c_1, c_2 sont des facteurs de poids, r_1, r_2 sont des nombres aléatoires entre [0 1], $pbest_o$ est la meilleure solution obtenue par la particule o, $gbest$ est la meilleure solution obtenue par toutes les particules.

Dans PSO, le déplacement d'une particule est influencé par les trois composantes suivantes :

— Une composante d'inertie wv_o^t : la particule tend à suivre sa direction courante de déplacement.

— Une composante cognitive $c_1r_1(pbest_o - x_o^t)$: la particule tend à se diriger vers le meilleur site par lequel elle est déjà passée ;

— Une composante sociale $c_2r_2(gbest - x_o^t)$: la particule tend à se fier à l'expérience de ses congénères et, ainsi, à se diriger vers le meilleur site déjà atteint par ses voisins.

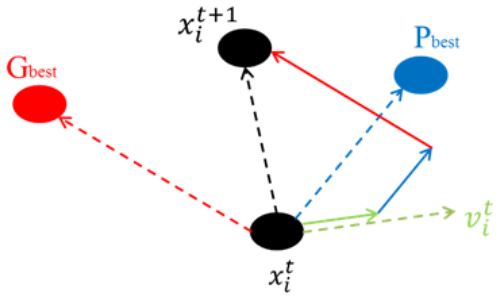

FIGURE 3.2 – Mouvement d'une particule dans l'espace de recherche

La stratégie de déplacement d'une particule est illustrée dans la figure 3.2.

3.5.3 Fonctionnement de PSO

L'algorithme de base de PSO est :

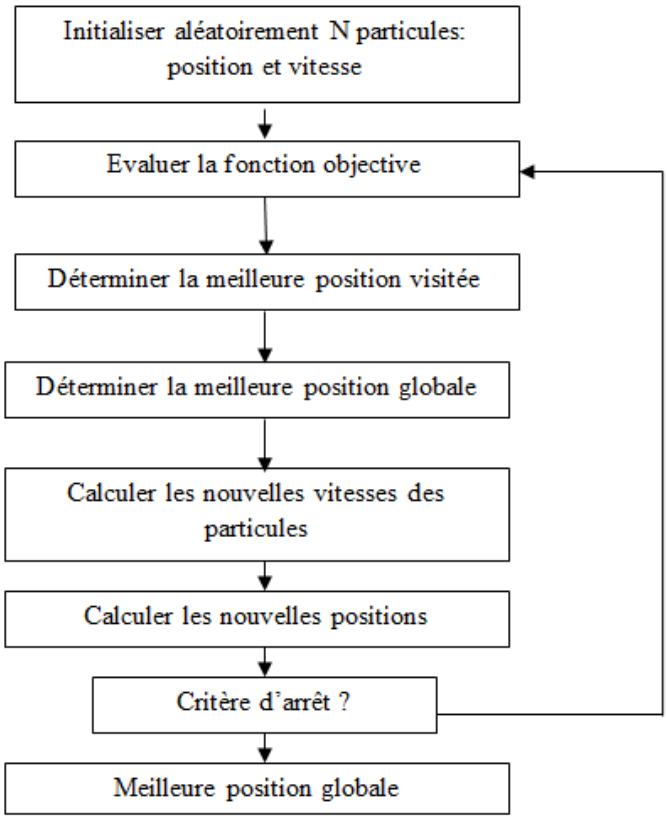

FIGURE 3.3 – Algorithme général de PSO.

3.6 Applications des métaheuristiques au problème de transport flou à quatre indices

Dans cette section, on va présenter deux algorithmes approchés afin de résoudre le problème de transport flou à quatre indices. Le premier est une application de l'algorithme génétique et le deuxième est une hybridation entre l'algorithme génétique et l'algorithme d'optimisation par essaim particulaire.

L'idée de ces deux algorithmes proposés est de déterminer les liens entre les destinations les plus prioritaires.

3.6.1 Codage d'une solution d'un PTF4

L'application des méthodes d'optimisation aura pour objectif de trouver la variable $\tilde{x}$ optimale. L'un des inconvénients de l'utilisation de méthodes métaheuristiques est qu'elles ne respectent pas les contraintes (solution hors domaine de recherche).

La contrainte $A\tilde{x} =_{\Re} \tilde{b}$ assure qu'il y n'a pas de surcharge c.-à-d. on doit respecter la capacité de chaque dimension i, j, k et l. Afin de garantir la vérification des contraintes, on doit pénaliser toute solution qui ne les respecte pas par affecter une valeur très grande à sa fonction objectif. Mais ça reste une méthode non-pratique.

Pour surmonter ce problème, on présente une méthode de calcul des fonctions objectifs qui assure la validité des contraintes implicitement.

En s'inspirant de la méthode exacte, on cherche l'importance de la case (i, j, k, l).

Soit V un vecteur de taille N ; ses éléments sont des réels donnés aléatoirement, on peut classer ces éléments par ordre croissant (ou décroissant). À partir de ce classement, on peut déterminer les indices (i, j, k, l) correspondants et leurs priorités. Donc on aura un ensemble des cases classés de 1 jusqu'à N.

$$(i_1, j_1, k_1, l_1), (i_2, j_2, k_2, l_2), \ldots, (i_N, j_N, k_N, l_N)$$

Example 3.1. Soit V un vecteur de 16 réels donnés aléatoirement dans l'intervalle $[0, 1]$, on va classer ces éléments par ordre croissant.

Vecteur continu	Vecteur discret (sort)	Quadruplet correspondant	Priorité
0.95	16	$(2, 2, 2, 2)$	1
0.2	5	$(1, 1, 2, 1)$	2
0.75	12	$(2, 2, 1, 2)$	3
0.4	9	$(1, 1, 1, 2)$	4
0.65	11	$(1, 2, 1, 2)$	5
0.1	3	$(1, 2, 1, 1)$	6
0.15	4	$(2, 2, 1, 1)$	7
0.25	6	$(2, 1, 2, 1)$	8
0.35	8	$(1, 2, 2, 1)$	9
0.6	10	$(2, 2, 2, 1)$	10
0.8	13	$(1, 1, 2, 2)$	11
0.85	14	$(2, 1, 2, 2)$	12
0.3	7	$(1, 2, 2, 1)$	13
0.9	15	$(1, 2, 2, 2)$	14
0.05	2	$(2, 1, 1, 1)$	15
0	1	$(1, 1, 1, 1)$	16

On détermine la quantité $\tilde{x}_{i_1,j_1,k_1,l_1}$ à affecter à la case numéro 1 (i_1, j_1, k_1, l_1)

$\tilde{x}_{i_1j_1k_1l_1} = \min(\tilde{\alpha}_{i_1}, \tilde{\beta}_{j_1}, \tilde{\gamma}_{k_1}, \tilde{\delta}_{l_1})$. Ensuite, on fait la mise à jour des données comme suit :

$$\begin{cases} \tilde{\alpha}_{i_1} = \tilde{\alpha}_{i_1} \ominus \tilde{x}_{i_1j_1k_1l_1} \\ \tilde{\beta}_{j_1} = \tilde{\beta}_{j_1} \ominus \tilde{x}_{i_1j_1k_1l_1} \\ \tilde{\gamma}_{k_1} = \tilde{\gamma}_{k_1} \ominus \tilde{x}_{i_1j_1k_1l_1} \\ \tilde{\delta}_{l_1} = \tilde{\delta}_{l_1} \ominus \tilde{x}_{i_1j_1k_1l_1} \end{cases}$$

et on répète la même opération jusqu'à la détermination de la quantité $\tilde{x}_{(i_N,j_N,k_N,l_N)}$ à affecter à la case (i_N, j_N, k_N, l_N) numéro N.

A la fin on va calculer la valeur de la fonction objectif associée.

Remarque 3.6.1. *Signalons que cette méthode de codage fournit à chaque fois une solution qui reste toujours dans le domaine de recherche (à la fin, on trouve que toutes les contraintes sont vérifiées) :*

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q}\tilde{x}_{ijkl} =_{\Re} \sum_{i=1}^{m}\tilde{\alpha}_i =_{\Re} \sum_{j=1}^{n}\tilde{\beta}_j =_{\Re} \sum_{k=1}^{p}\tilde{\gamma}_k =_{\Re} \sum_{l=1}^{q}\tilde{\delta}_l.$$

3.6.2 Algorithme génétique pour le PTF4

L'algorithme génétique proposé est le suivant :

Entrée — Fixer la taille de population N_p.
- — Fixer le nombre de générations final N_g.
- — Lire les data du problème : $m, n, p,$ et q, la matrice des coûts $\tilde{c}$ et les quantités $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}$ et $\tilde{\delta}$.

Initialisation de la population On crée N_p individus, Chaque individu est un vecteur de N éléments.

Itération $t = 1$

Tant que : ($t < N_g$) faire :

Evaluation de la population

Pour $cpt = 1 : N_p$ **faire :**

Evaluer la fonction d'objectif de chacun des N_p individus.

Fin pour.

Sélection { Choisir aléatoirement $\frac{N_p}{2}$ individus. L'individu qui correspond à la plus petite valeur d'objectif doit être choisi parmis eux.

Croisement

Appliquer le croisement en 3-points.

Mutation

$\left\{\begin{array}{l}\text{La probabilité de la mutation est } 0.1\\ \text{On choisit deux nombres, le premier entre } 0 \text{ et } 1 \text{ et le deuxième entre } 1 \text{ et } \frac{N_p}{2}.\\ \text{Si le premier est} \leq 0.1, \text{ l'enfant correspondant au deuxième numéro choisi}\\ \text{doit être changé aléatoirement.}\end{array}\right.$

Nouvelle génération Les parents et leurs enfants.

$t = t + 1$

Fin tant que.

3.6.3 Algorithme PSO-GA pour le PTF4

Principe PSO-GA

Plusieurs articles qui ont étudié les algorithmes génétiques ont montré ses capacité dans l'exploitation. Toutefois, il y a d'autres articles [5] qui ont mentionné PSO pour ses performances dans la phase d'exploration. Dans ce qui suit, on va présenter une autre méthode d'optimisation faisant appel à l'hybridation entre PSO et GA. L'idée de base de cette méthode est de combiner le comportement social de PSO (Gbest) avec la recherche locale donnée par GA.

Dans l'algorithme PSO-GA, on va adapter le même principe pour déterminer les priorités des cases de tableau de transport. L'amélioration introduite ici est l'ajoute des variable (Z_o) qui servent à contrôler la quantité transportée. Ce contrôle minimise ou bien limite les quantités transportés de chaque case. Les vecteurs $Z_o;\ o = 1 : N$ seront mis à jour par l'algorithme de PSO.

Algorithme PSO-GA pour le PTF4

L'algorithme PSO-GA proposé est le suivant :

Entrée — Fixer la taille de population N_p.
- — Fixer le nombre de générations final N_g.
- — Lire les données du problème : $m, n, p,$ et q, la matrice des coûts $\tilde{c}$ et les quantités $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}$ et $\tilde{\delta}$.
- — Initialiser les valeurs de $c_1, c_2 = 1.5$ et $w = 0.7$.

Initialisation de la population On crée N_p individus ; Chaque individu est un vecteur de N éléments.

Initialisation de vitesses v et positions des particules Z de la même manière que l'initialisation de population.

Itération $t = 1$

Tant que : ($t < N_g$) **faire** :

Evaluation de la population

Pour $cpt = 1 : N_p$ **faire :**

Evaluer la fonction d'objectif de chacun des N_p individus et N_p particules

Prendre $\tilde{x}_{ijkl} = \min(\tilde{\alpha}_i, \tilde{\beta}_j, \tilde{\gamma}_k, \tilde{\delta}_l, Z_{cpt})$

Mis à jour $pbest_{cpt}$

Fin pour.

Mis à jour $gbest$

Sélection { Choisir aléatoirement $\frac{N_p}{2}$ individus. L'individu qui correspond à la plus petite valeur d'objectif doit être choisi parmis eux.

Croisement Appliquer le croisement en 3-points.

Mutation { La probabilité de la mutation est 0.1 On choisit deux nombres, le premier entre 0 et 1 et le deuxième entre 1 et $\frac{N_p}{2}$. Si le premier est $\leq 0.1,$ l'enfant correspond au deuxième numéro choisi doit être changé aléatoirement.

La nouvelle génération Les parents et leurs enfants.

Mise à jour de vitesses et positions des particules

Pour $cpt = 1 : N_p$ **faire :**

$$v_{cpt}^{t+1} = w \times v_{cpt}^{t} + r_1 \times c_1 \times (pbest_{cpt}^{t} - Z_{cpt}^{t}) + r_2 \times c_2 \times (gbest - Z_{cpt}^{t})$$

$$Z_{cpt}^{t+1} = Z_{cpt}^{t} + v_{cpt}^{t+1}$$

Fin pour.

$t = t + 1$

Fin tant que

3.7 Etude numérique comparative

Dans cette section, nous allons tester les performances des algorithmes précédents (exacte, GA et PSO-GA) en prenant un échantillon de 1000 problèmes PTF4 de différentes tailles, to-

talement flous et dont les data sont des nombres flous trapézoïdaux. Signalons que nos programmes sont écrits en Matlab et implémentés sur une machine I3 sous l'environnement Microsoft Windows.

Définition 3.7.1. *On appelle "efficacité" d'une méthode approchée vis-à-vis à une méthode exacte (considérée comme Benchmark) l'expression suivante :*

$$\left(1 - |\frac{V_{opt} - V_{app}}{V_{opt}}|\right) \times 100$$

Où, V_{opt} est la valeur de l'objectif donnée par la méthode exacte et V_{app} est la valeur de l'objectif donnée par la méthode approchée.

Dans les tableaux suivants, nous désignons par :

- V_opt : la valeur optimale obtenue par la méthode exacte.
- V_GA : la valeur finale obtenue par l'algorithme génétique.
- V_PSO-GA : la valeur finale obtenue par l'algorithme hybride PSO-GA.
- $M * N$: la taille du problème, où $M = m + n + p + q$ et $N = mnpq$.
- $rank(\cdot)$: la fonction de classement d'un nombre flou.
- T_M-ext : le temps d'exécution occupé par la méthode exacte en millisecondes.
- T_PSO-GA : le temps d'exécution occupé par l'algorithme hybride PSO-GA en millisecondes.
- T_GA : le temps d'exécution occupé par l'algorithme génétique en millisecondes.

Taille	Méthode exacte			Algorithme Génétique			Algorithme hybride PSO - GA		
N*M	V_opt	Rank (V_opt)	T_M-exacte	V_GA	Rank (V_GA)	T_GA	V_PSO-GA	Rank (V_PSO-GA)	T_PSO-GA
16*8	[-6.550700e+03 - 1.310140e+03 5.240560e+03 1.310140e+04]	**6550,7**	**23,182608**	[-6.550700e+03 - 2.183567e+03 4.367133e+03 1.310140e+04]	**6550,7**	**20,00101**	[-2.407606e+03 - 1.313240e+03 - 1.094366e+03 2.626479e+04]	**6566,1984**	**24,001218**
10*36	[-3.960150e+03 - 2.640100e+03 - 1.320050e+03 2.112080e+04]	**5280,2**	**23,187231**	[-4.400167e+03 - 2.640100e+03 - 1.760067e+03 2.112080e+04]	**5280,2**	**20,00232**	[-1.323110e+03 - 8.820734e+02 3.087257e+03 1.587732e+04]	**5292,4405**	**24,002785**
11*48	[-6692 - 6.692000e+02 - 6.692000e+02 1.606080e+04]	**2676,8**	**23,190391**	[-4.015200e+03 - 8.922667e+02 - 4.461333e+02 13384]	**2676,8**	**20,00311**	[-6.719703e+02 - 4.479802e+02 2.687881e+03 6.943693e+03]	**2687,8812**	**24,003734**
12*81	[-2.725600e+03 - 9.085333e+02 2.725600e+03 4.542667e+03]	**2725,6**	**23,200111**	[-4.093540e+03 - 7.223893e+02 - 4.815929e+02 1.444779e+04]	**2889,5574**	**20,00531**	[-1.366456e+03 - 5.465823e+02 3.552785e+03 5.465823e+03]	**2732,9117**	**24,006368**
14*144	[-3.064150e+03 - 1.225660e+03 6.128300e+03 1.409509e+04]	**6128,3**	**23,22163**	[-1.225660e+03 - 1.021383e+03 1.000956e+04 1.225660e+04]	**6128,3**	**20,00954**	[-9.222107e+03 - 1.537018e+03 - 1.537018e+03 3.074036e+04]	**6148,0715**	**24,011451**
16*256	[-3.698807e+03 - 1.849403e+03 - 1.849403e+03 1.479523e+04]	**3698,8067**	**23,266294**	[-7.693084e+02 - 6.410904e+02 6.282686e+03 7.693084e+03]	**3846,5422**	**20,01716**	[-9.620470e+03 - 1.282729e+03 - 6.413647e+02 2.308913e+04]	**3848,1879**	**24,020594**
18*400	[-2.822475e+04 - 1.128990e+04 - 5.644950e+03 5.644950e+04]	**11289,9**	**23,332031**	[-3.612768e+04 - 1.128990e+04 - 2.257980e+03 6.773940e+04]	**11289,9**	**20,02706**	[-9.409578e+03 - 5.645747e+03 - 3.763831e+03 4.516598e+04]	**11291,494**	**24,032466**

FIGURE 3.4 – Extrait de la base de données (petite taille)

Taille	Méthode exacte			Algorithme Génétique			Algorithme hybride PSO - GA		
N*M	V_opt	Rank (V_opt)	T_M-exacte	V_GA	Rank (V_GA)	T_GA	V_PSO-GA	Rank (V_PSO-GA)	T_PSO-GA
20*625	[-1.853500e+03 - 9.267500e+02 1.853500e+03 2.780250e+03]	**1853,5**	**23,44805**	[-3.089167e+03 - 9.267500e+02 - 3.089167e+02 9.267500e+03]	**1853,5**	**20,04265**	[-1.868771e+03 - 3.114619e+02 1.868771e+03 3.426081e+03]	**1868,7714**	**24,0512**
22*900	[-1.617436e+04 - 4.411189e+03 - 2.940792e+03 4.411189e+04]	**8822,3775**	**23,60627**	[-9.110603e+03 - 2.277651e+03 6.832952e+03 1.822121e+04]	**9110,6029**	**20,06187**	[-1.110276e+04 - 8.882206e+03 - 2.220551e+03 3.552882e+04]	**8882,2059**	**24,0742**
24*1296	[-4.428258e+02 - 4.428258e+02 8.856517e+02 5.313910e+03]	**1736,7446**	**23,85821**	[-3.107784e+03 - 2.071856e+03 - 1.035928e+03 8.287425e+03]	**2071,8562**	**20,08974**	[-2.894574e+03 - 8.683723e+02 - 2.894574e+02 8.683723e+03]	**1771,3033**	**24,1077**
32*4096	[-1.506359e+03 - 1.506359e+03 9.038153e+03 2.410174e+04]	**9038,1529**	**26,12482**	[-7.114900e+03 - 4.743267e+03 - 2.371633e+03 3.794614e+04]	**9486,5338**	**20,29024**	[-9.041058e+03 - 4.520529e+03 4.520529e+03 1.808212e+04]	**9041,0575**	**24,3483**
40*10000	[-1.772005e+03 - 3.544011e+02 - 3.544011e+02 5.316016e+03]	**1772,0053**	**32,40356**	[-1.156758e+03 - 5.783790e+02 2.891895e+03 4.627032e+03]	**2313,516**	**20,72136**	[-9.304405e+02 - 6.202937e+02 1.860881e+03 4.031909e+03]	**1860,881**	**24,8656**
50*22500	[-5.248124e+03 - 9.048489e+02 - 7.238791e+02 1.809698e+04]	**3555,2645**	**49,25362**	[-9.930391e+02 - 7.944313e+02 6.156843e+03 7.944313e+03]	**3972,1565**	**21,6496**	[-1.777632e+03 - 1.185088e+03 5.925441e+02 1.066579e+04]	**3619,3957**	**25,9795**
60*50625	[-1.056424e+04 - 2.112848e+03 - 2.112848e+03 3.169271e+04]	**10564,238**	**97,00571**	[-1.467890e+04 - 1.100918e+04 - 3.669726e+03 4.403671e+04]	**11009,179**	**23,77228**	[-5.306705e+03 - 2.653352e+03 1.061341e+04 2.388017e+04]	**10613,409**	**28,5267**

FIGURE 3.5 – Extrait de la base de données (moyenne taille)

Taille	Méthode exacte			Algorithme Génétique			Algorithme hybride PSO - GA		
N*M	V_opt	Rank (V_opt)	T_M-exacte	V_GA	Rank (V_GA)	T_GA	V_PSO-GA	Rank (V_PSO-GA)	T_PSO-GA
70*90000	[-6.128768e+03 -3.064384e+03 -3.064384e+03 1.225754e+04]	**3003,5827**	**177,8124**	[-6.922209e+02 -5.768507e+02 2.192033e+03 1.038331e+04]	**3461,10436**	**26,783887**	[-7.508957e+02 -6.007165e+02 4.655553e+03 6.007165e+03]	**3064,3841**	**32,14066449**
80*1,6e+05	[-2.122077e+03 -1.061038e+03 -1.061038e+03 6.366231e+03]	**2068,4328**	**347,28859**	[-1.281875e+03 -8.545831e+02 2.991041e+03 5.127499e+03]	**2563,74943**	**32,199826**	[-3.102649e+03 -2.068433e+03 -1.034216e+03 8.273731e+03]	**2122,0769**	**38,63979107**
90*2,5e+05	[-8.875769e+03 -1.868583e+03 -1.401437e+03 2.802874e+04]	**5605,7487**	**598,84652**	[-1.331647e+04 -6.146065e+03 -1.024344e+03 3.073032e+04]	**6146,06468**	**39,233134**	[-5.619608e+03 -1.873203e+03 -1.873203e+03 1.685883e+04]	**5619,6084**	**47,0797609**
100*390625	[-2.225178e+04 -7.026878e+03 -1.171146e+03 4.216127e+04]	**7026,8785**	**1046,0201**	[-2.037742e+04 -3.773597e+03 -1.509439e+03 4.528317e+04]	**7547,19437**	**50,321207**	[-7.077615e+03 -2.359205e+03 -2.359205e+03 2.123285e+04]	**7077,6152**	**60,38544844**
200*6,25e+06	[-4.576872e+03 -2.615355e+03 -1.961516e+03 3.138426e+04]	**7846,0656**	**36639,409**	[-2.763482e+03 -1.658089e+03 8.290447e+03 2.044977e+04]	**8290,44682**	**532,80101**	[-2.621544e+03 -1.310772e+03 3.932316e+03 2.359390e+04]	**7864,6318**	**639,3612066**
400*1,0e+08	[-2.524400e+03 -4.207333e+02 -4.207333e+02 7.573200e+03]	**2524,3999**	**1322690,3**	[-2.140276e+03 -1.528769e+03 -6.115075e+02 1.223015e+04]	**3057,53738**	**8692,6386**	[-1.144772e+03 -6.359842e+02 -5.087874e+02 1.017575e+04]	**2543,9369**	**10431,16635**
800*1,6e+09	[-1.099250e+04 -4.711070e+03 -1.570357e+03 2.355535e+04]	**4706,0605**	**48066771**	[-3.073281e+03 -1.756161e+03 -1.317120e+03 2.107393e+04]	**5268,48183**	**146694,17**	[-4.706061e+03 -2.353030e+03 -2.353030e+03 1.411818e+04]	**4711,0696**	**176033,0013**

TABLE 3.6 – Extrait de la base de données (grande taille)

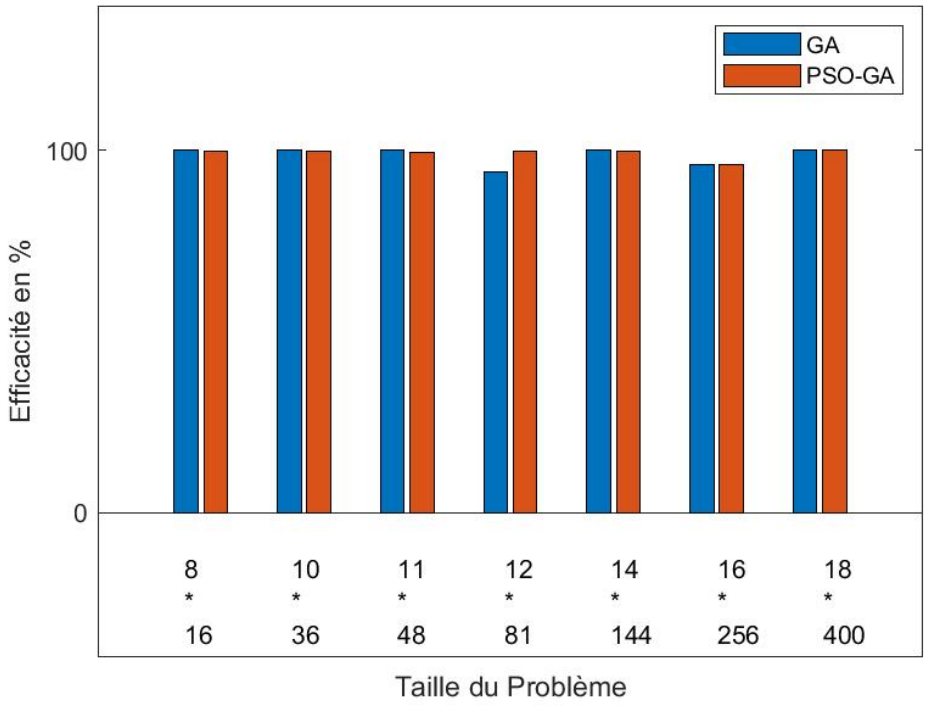

FIGURE 3.7 – Efficacité des algorithmes pour des problèmes de petite taille

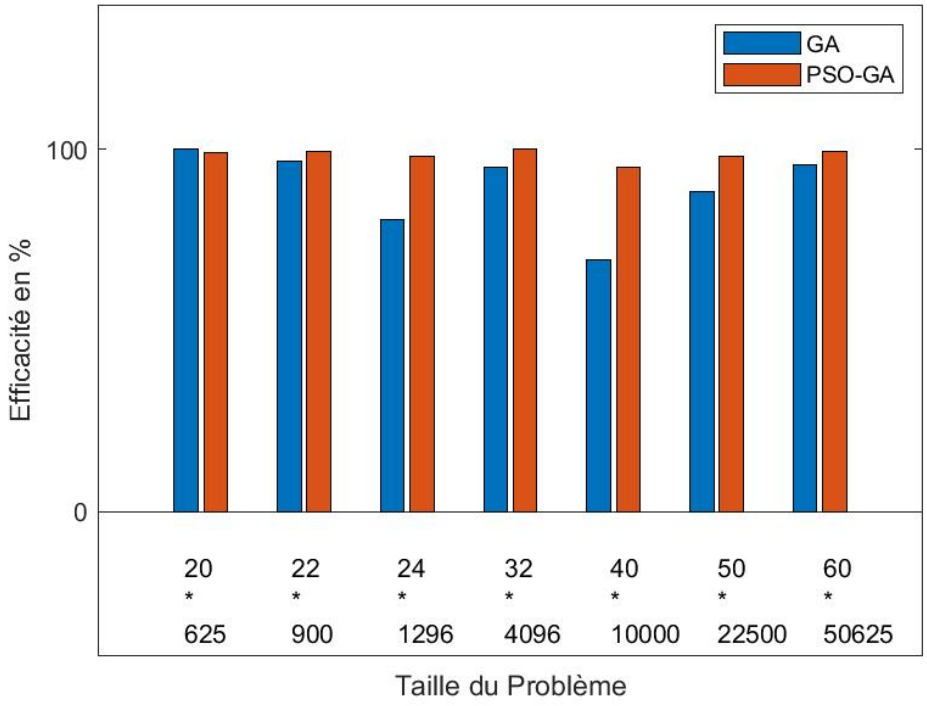

FIGURE 3.8 – Efficacité des algorithmes pour des problèmes de moyenne taille

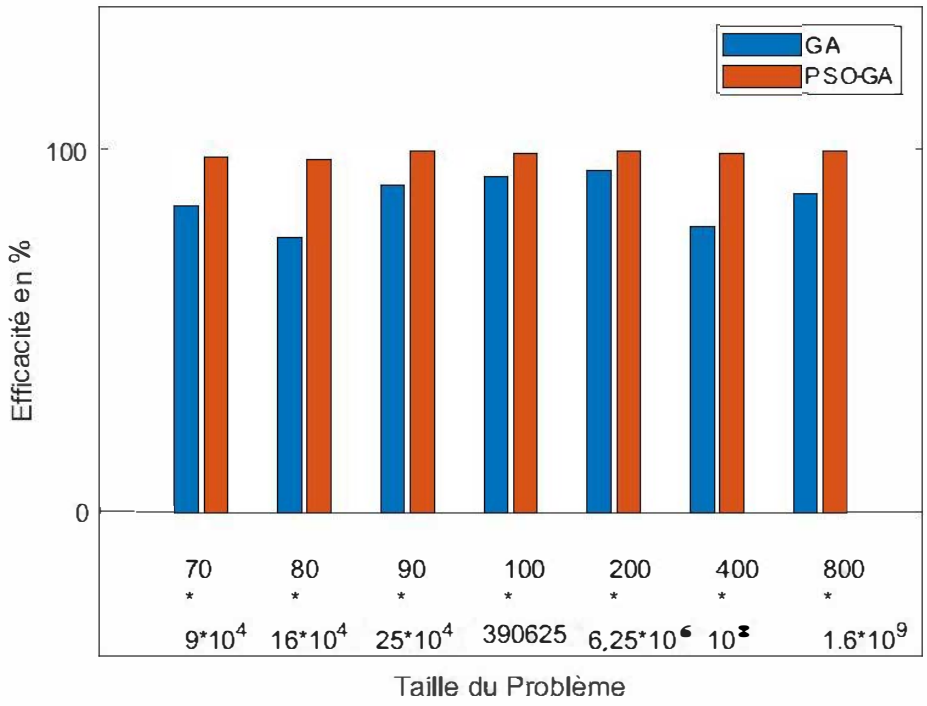

FIGURE 3.9 – Efficacité des algorithmes pour des problèmes de grande taille

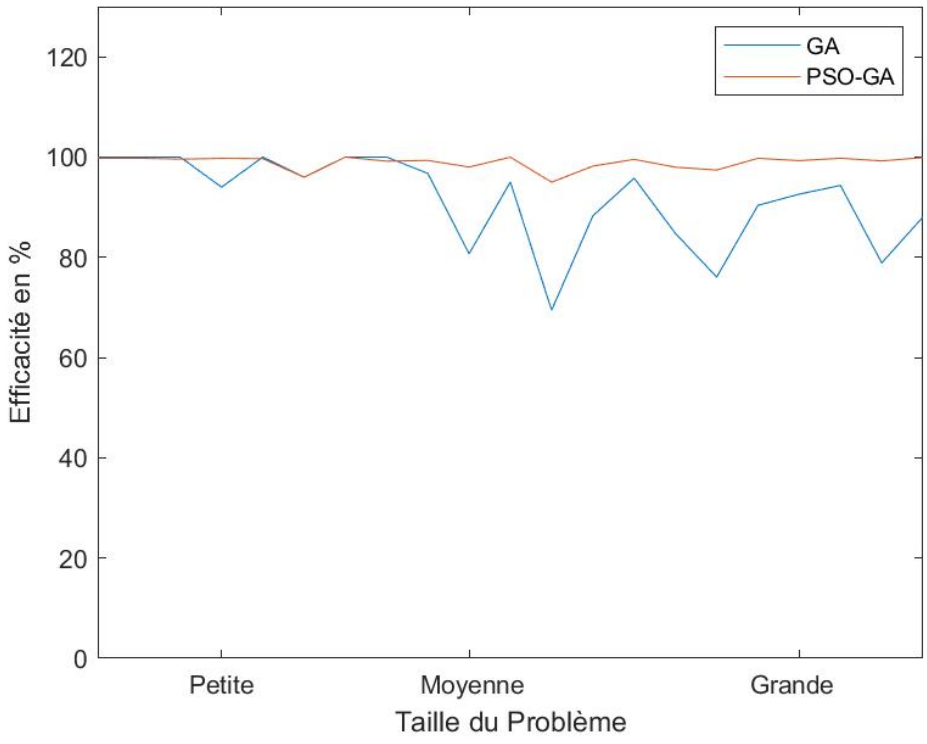

FIGURE 3.10 – Courbes comparatives de l'efficacité en fonction de la taille du problème

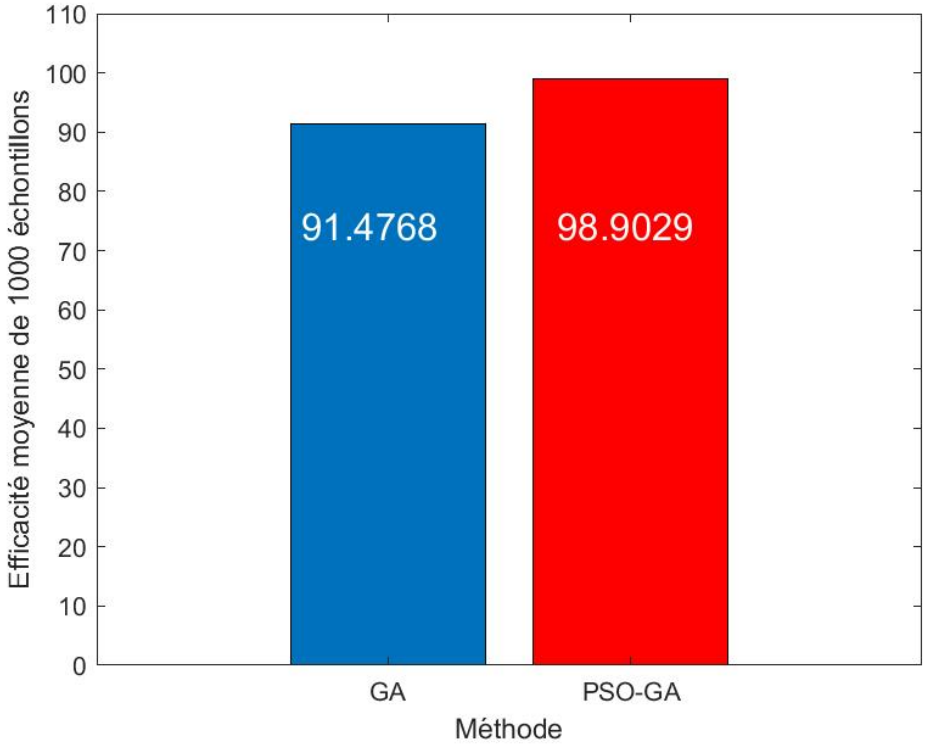

FIGURE 3.11 – Moyenne de l'efficacité des algorithmes appliqués

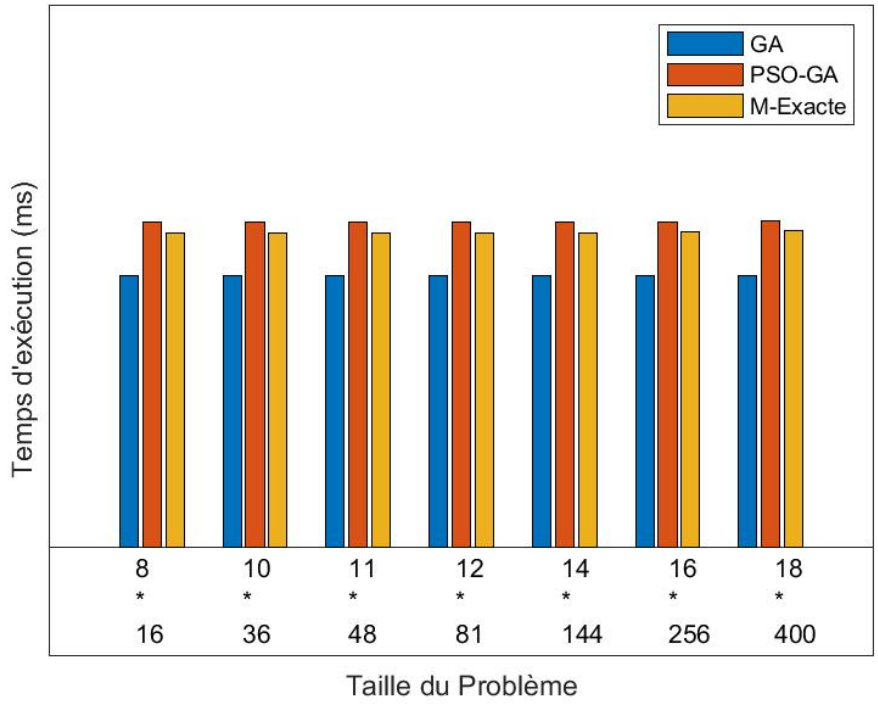

FIGURE 3.12 – Temps d'exécution des algorithmes pour des problèmes de petite taille

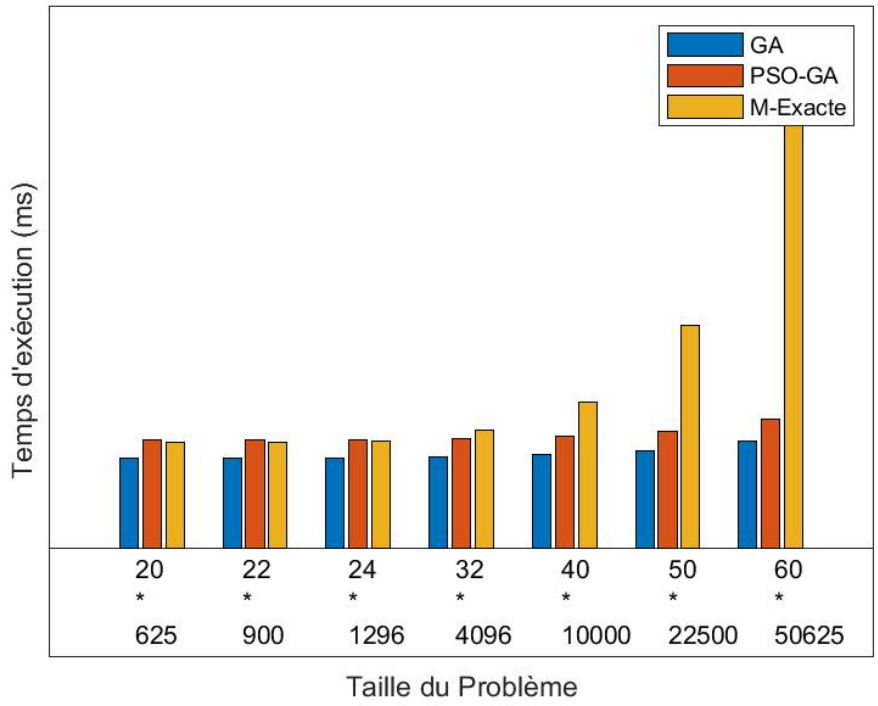

FIGURE 3.13 – Temps d'exécution des algorithmes pour des problèmes de moyenne taille

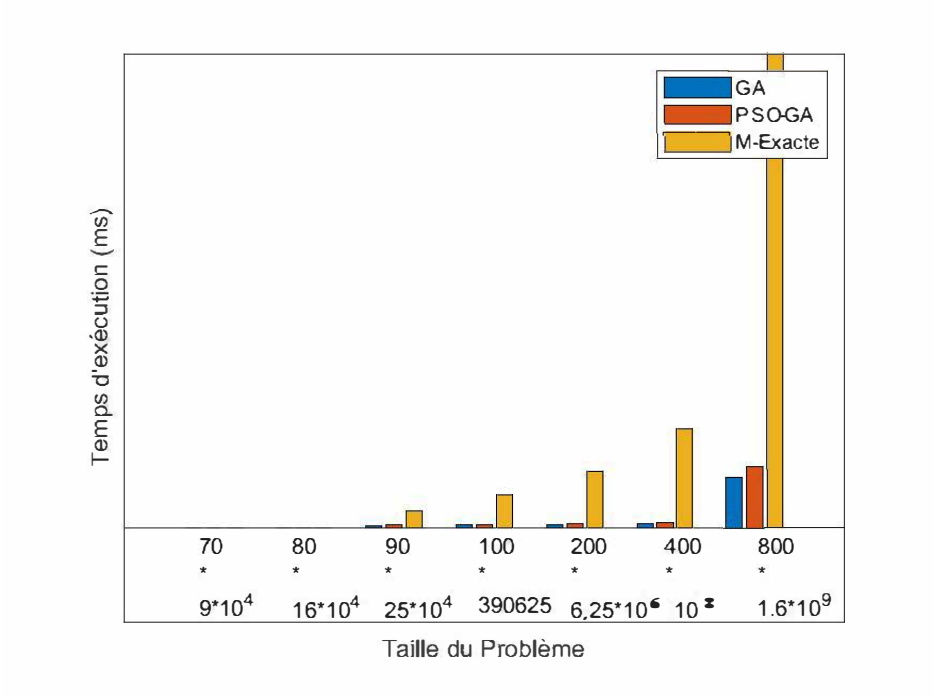

FIGURE 3.14 – Temps d'exécution des algorithmes pour des problèmes de grande taille

Commentaires

Les résultats obtenus montrent :

— La supériorité de l'algorithme hybride PSO-GA que nous avons élaboré pour résoudre les PTF4 de grande taille.

— L'algorithme hybride PSO-GA donne en général, une solution proche de l'optimum (et parfois optimale).

— Lorsque la taille du problème est relativement petite, le temps d'exécution de l'algorithme de transport (exacte) est mieux que celui de GA ou de PSO-GA. Par contre dans le cas de taille moyenne ou grande, ces deux derniers algorithmes fournissent des solutions au PTF4 plus rapidement que l'algorithme de transport.

— Dans la plupart des exemples traités, l'algorithme PSO-GA fournit la solution plus rapidement que le GA, ceci revient au bon compromis entre l'exploration et l'exploitation de l'algorithme hybride.

— Contrairement à la méthode exacte, les deux algorithmes approchés ne présentent pas de phénomène de dégénérescence.

— De nos différents tests numériques établis, on peut conclure que l'algorithme hybride PSO-GA est efficace pour la résolution du PTF4 de tailles différentes et souvent fournit une solution proche de l'optimum dans un temps considérablement réduit.

3.8 Conclusion

Dans ce chapitre, nous avons résolu le problème de transport flou à quatre indices en utilisant deux méthodes métaheuristiques. Notre choix s'est porté sur les algorithmes d'optimisation à base de population à savoir PSO et GA vu qu'ils ont eu un grand succès pour de nombreux problèmes pratiques. L'algorithme PS0-GA exploite les avantages de ces deux algorithmes, ceci a été clairement prouvé par nos résultats de simulation obtenus.

Conclusion générale et perspectives

Dans cette thèse, nous nous sommes intéressés à la résolution d'un problème de transport flou à quatre indices (PTF4) par deux types de méthodes : exactes et approchées. Dans un premier lieu, nous avons introduit une adaptation aux méthodes de coût minimum, d'approximation de Russell et d'approximation de Vogel pour établir une solution de base réalisable initiale au PTF4. Pour déterminer la solution optimale floue, nous avons adapté également la phase 2 de l'algorithme AL_{PT4C} introduit par R. Zitouni et al. (2003) [44] au contexte flou.

Les tests numériques effectués sur de nombreux problèmes sous forme PTF4 à différentes tailles, approuvent l'efficacité connue dans le cas net de ces algorithmes même pour ce type de problèmes.

L'application des méthodes métaheuristiques à la résolution des PTF4 vient dans l'objectif de traiter l'inconvénient de temps d'exécution coûteux que présente l'algorithme exact dans le cas des problèmes de grandes tailles. Notre choix s'est porté sur l'Algorithme Génétique (en anglais "Genetic Algorithm" (GA)) et celui de l'Optimisation par essaims particulaires qui porte le nom "Particle Swarm Optimization (PSO)" en anglais, vu à leur réputation dans la littérature. Initialement, nous avons proposé une méthode de codage d'une solution du PTF4. En appliquant l'algorithme GA au PTF4, nous avons constaté que l'amélioration du temps d'exécution vient au coût de la qualité de la solution finale. En combinant les avantages de ces deux algorithmes (l'exploitation de GA et l'exploration de PSO), nous avons pu introduire un nouvel algorithme hybride pour résoudre le PTF4. D'après nos résultats numériques, nous constatons que cet algorithme et fournit souvent une solution proche de la solution optimale dans un temps considérablement réduit, surtout pour les problèmes de taille relativement grande et qu'il présente un bon compromis entre vitesse et précision ce qui montre son efficacité et robustesse.

Notons que tous les résultats obtenus sont indépendants du nombre d'indices, ce qui prouve que ces méthodes peuvent être généralisées pour résoudre les problèmes de transport flous à indices multiples.

Perspectives

Nos perspectives peuvent être résumées dans les points suivants :

1. Généraliser les méthodes étudiées dans cette thèse pour résoudre les problèmes de transport flous à indices multiples.
2. Proposer d'autres méthodes hybrides entre des méthodes exactes et métaheuristiques au problème de transport flou à quatre indices.
3. Résoudre les problèmes de transport non balancés à n-indices où $n \geq 3$.

Bibliographie

[1] D. Aaid, Étude numérique comparative entre des méthodes de résolution d'un problème de transport à quatre indices avec capacités, Thèse de Magister en Mathématiques, Faculté de Mathématiques, l'Université de Constantine, (2010).

[2] A. E. M. Abd Elazeem, A. A. A. Mousa, M. A. El-Shorbagy, S. K. Elagan and Y. Abo-Elnaga, Detecting All Non-Dominated Points for Multi-Objective Multi-Index Transportation Problems, Sustainability, 13, 1372, (2021).

[3] M. Achache, A polynomial-time weighted path-following interior-point algorithm for linear optimization, Asian-European Journal of Mathematics, 13, 2050038, (2020).

[4] M. Achache, N. Hazzam, Solving absolute value equations via complementarity and interior point methods, J. Nonl. Funct. Anal, 1-10, (2018).

[5] A. Baykaso$\tilde{g}$lu, K. Subulan, A direct solution approach based on constrained fuzzy arithmetic and metaheuristic for fuzzy transportation problems, Soft Computing, 23(5), 1667-1698, (2019).

[6] S. Bourazza, Variantes d'algorithmes génétiques appliquées aux problèmes d'ordonnancement, Thèse Doctorat de l'Université du Havre, (2006).

[7] H. Bulut and S.A. Bulut, Construction and algebraic characterizations of a planar four-index transportation problem equivalent to a circularization network flow problem, Int. J. Comput. Math., 80, 1373-1383, (2003).

[8] S. Chanas, W. Kolodziejczyk, A. Machaj, A fuzzy approach to the transportation problem, Fuzzy Sets Syst, 13, 211-221, (1984).

[9] S. Chanas and D. Kuchta, A concept of the optimal solution of the transportation problem with fuzzy cost coefficients. Fuzzy Sets Syst, 82, 299-305, (1996).

[10] A. L. Custódio, L. N. Vicente, Using Sampling and Simplex Derivatives in Pattern Search Methods, Siam journal on optimization, 18, 537-555, (2007).

[11] G. B. Dantzig, Application of the Simplex Method to a Transportation Problem, Activity Analysis of Production and Allocation, Koopmans, T.C., Ed., John Wiley and Sons, 359-373, (1951).

[12] A. Ebrahimnejad, J. L. Verdegay, A new approach for solving fully intuitionistic fuzzy transportation problems, Fuzzy Optimization and Decision Making, 17, 447-474, (2017).

[13] A. Gani, K. A. Razak, Two stage fuzzy transportation problem. J. Phys. Sci., 10, 63-69, (2006).

[14] H. Hachimi, Hybridations d'Algorithmes Métaheuristiques en Optimisation Globale et Leurs Applications , Thèse de Doctorat en Optimisation, Analyse Numérique, Statistique, Institut National des Sciences Appliquées de Rouen, (2013).

[15] K. B. Haley, The multi-index transportation problem, Oper. Res., 11, 368-379, (1963).

[16] M. Hedid, R. Zitouni, Solving the four-index fully fuzzy transportation problem. Croatian Operational Research Review, 11, 199-215, (2020).

[17] M. Hedid, R. Zitouni, S. Labdai, Adaptive algorithm for solving the four-index fuzzy transportation problem. IEEE 6th International Conference on Optimization and Applications, ICOA 2020, 1-5, (2020).

[18] F. L. Hitchcock, The distribution of a product from several sources to numerous localities, Jour. Math. Phys., 20, 224-230, (1941).

[19] N. K. Karmarkar, A new polynomial-time algorithm for linear programming, in : Proceedings of the 16th Annual ACM Symposium on Theory of Computing, 302-311, (1984).

[20] A. Kaur, A. Kumar, A new approach for solving fuzzy transportation problems using generalized trapezoidal fuzzy numbers. Appl. Soft Comput. 12, 1201-1213, (2012).

[21] D. Kaur, S. Mukherjee, and K. Basu, Solution of a multi-objective and multi-index real-life transportation problem using different fuzzy membership functions, J. Optim. Theory Appl, 164, 666-678, (2015).

[22] J. Kennedy et R. Eberhart, Particle Swarm Optimization, Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, 1942-1948, (1995).

[23] A. Kumar, A. Kaur, A new method for solving fuzzy transportation problems using ranking function, Appl. Math. Model, 35, 5652-5661, (2011).

[24] A. Lemouari, Introduction aux métaheuristique, Support de cours, spétialité (Système d'information et aide à la prise de décision, Intelligence Artificielle), Université de Jijel, (2014).

[25] S. T. Liu, C. Kao, Solving fuzzy transportation problems based on extension principle, Eur. J. Oper. Res, 153, 661-674, (2004).

[26] E. H. Mamdani, S. Assilian, An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller, Int. J. Man-Machine Studies, 7, 1-13, (1975).

[27] M. Oheigeartaigh, A fuzzy transportation algorithm, Fuzzy Sets Syst, 8, 235-243, (1982).

[28] P. Pandian, and G. Natarajan, A new algorithm for finding a fuzzy optimal solution for fuzzy transportation problems, Appl. Math. Sci., 4, 79-90, (2010).

[29] T.-H. Pham, P. Dott, An exact method for solving the four index transportation problem and industrial application. Am. J. Oper. Res., 3, 28-44, (2013).

[30] J. Pratihar, R. Kumar, S. A. Edalatpanah, and A. Dey, Modified Vogel's approximation method for transportation problem under uncertain environment, Complex & Intelligent Systems, (2020).

[31] M. Queyranne, F. C. R. Spieksma, Approximation algorithms for multi-index transportation problems with decomposable costs, Discrete Appl. Math., 76, 239-253, (1997).

[32] D. S. Sarode, R. Tuli, Optimal Solution of the Planar Four Index Transportation Problem, In Proceedings of the 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates, 4-6 February 2019 ; IEEE : Piscataway Township, NJ, USA. (2019).

[33] P. Senthil Kumar, PSK method for solving type-1 and type-3 fuzzy transportation problems, International Journal of Fuzzy System Applications, 5, 121-146, (2016).

[34] P. Senthil Kumar, A simple method for solving type-2 and type-4 fuzzy transportation problems, International Journal of Fuzzy Logic and Intelligent Systems, 16, 225-237, (2016).

[35] P. Senthil Kumar, A Simple and Efficient Algorithm for Solving Type-1 Intuitionistic Fuzzy Solid Transportation Problems, International Journal of Operations Research and Information Systems, 9, 90-122, (2018).

[36] G. A. Vignaux, Z. Michalewicz, A genetic algorithm for the lineair transportation problem, IEEE transaction systèmes, Man and cybernetics, 21, (1991).

[37] S. K. Singh, S. P. Yadav, Efficient approach for solving type-1 intuitionistic fuzzy transportation problem, Int J Syst Assur Eng Manag, (2014).

[38] V. Skitsko, M. Voinikov, Solving four-index transportation problem with the use of a genetic algorithm. LogForum , 16, 397-408, (2020).

[39] K. Tanwar, S.K. Chauhan, Time-Cost Solution Pairs in Multi-index Bulk Transportation Problem, In International Conference on Recent Developments in Science, Engineering and Technology, Springer : Berlin/Heidelberg, Germany, (2019).

[40] L. Wang, A.S. Esenkov and A.P. Tizik, Torchinskaya, E.V. Decomposition Method for Solving Three-Index Transportation Problems, J. Comput. Syst. Sci. Int, 57, 759-765, (2018).

[41] L. Zadeh, Fuzzy sets, Information and control, 8, 338-353, (1965).

[42] H. J. Zimmermann, Fuzzy programming and linear programming with several objective function, Fuzzy Set and Systems, 1, 45-55, (1978).

[43] R. Zitouni, M. Achache, A numerical comparison between two exact methods for solving a capacitated 4-index transportation problem, Journal of Numerical Analysis and Approximation Theory, 46, 181-192, (2017).

[44] R. Zitouni, A. Keraghel, Resolution of a capacitated transportation problem with four subscripts. Kybernetes, 32, 1450-1463, (2003).

[45] R. Zitouni, A. Keraghel, A note on the algorithm of resolution of a capacitated transportation problem with four subscripts. Far East Journal of Mathematical Sciences (FJMS), 26, 769-778, (2007).

[46] R. Zitouni, A. Keraghel and D. Benterki, Elaboration and implantation of an algorithm solving a capacitated four-index transportation problem. Appl. Math. Sci., 1, 2643-2657, (2007).

[47] R. Zitouni, Etude qualitative de modèle de transport et localisation, Thèse de Doctorat de Mathèmatiques appliquées, Universitè Farhat Abbas, Sétif, (2007).

ملخص:

نهتم في هذه الأطروحة بحل مسألة النقل الضبابية ذات أربعة أدلة باستخدام طريقة دقيقة من جهة وطريقة تقريبية من جهة أخرى. نقدم في الجزء الأول تكيفًا في سياق ضبابي بأربعة أدلة للطرق الأكثر شيوعًا المستخدمة عادةً لحل المسألة الكلاسيكية مما يسمح بإيجاد الحل الغامض الأمثل بطريقة دقيقة.

في الجزء الثاني، نقترح طريقة تقريبية (Metaheuristic) **هجينة للحل وهي عبارة عن دمج بين طريقة** PSO **والخوارزميات الجينية** (GA). **هذه الطريقة موجهة بشكل أساسي لحل المسائل كبيرة الحجم أين تستغرق الطرق الدقيقة وقتًا طويلاً على العموم للوصول إلى الحل الأمثل. في الأخير، نقدم دراسة عددية لمقارنة فعالية ومتانة كلا الطريقتين.**

كلمات دلالية: **البرمجة الخطية، مسألة النقل الضبابي، مسألة النقل متعددة الأدلة، الخوارزميات الجينية، خوارزميات هجينة،** Metaheuristics**.**

Résumé :

Dans cette thèse, nous nous intéressons à la résolution du problème de transport flou à quatre indices par deux méthodes, l'une est exacte et l'autre est approchée. Nous présentons dans une première partie, une adaptation d'une méthode exacte introduite par Zitouni et al. pour résoudre un tel problème dans le cas net (par opposé au mot flou).

Ensuite, dans la deuxième partie, nous proposons une méthode approchée (Metaheuristique). Il s'agit d'une hybridation entre l'optimisation d'essaims particulaires (OEP) et les algorithmes génétiques (AG). Elle est adressée essentiellement à résoudre les problèmes de grande taille le cas où les méthodes exactes prennent en général, un temps considérable pour fournir une solution optimale. A la fin de ce document, nous donnons une étude numérique comparative afin de tester la robustesse et l'efficacité des deux approches.

Mots-Clés : Programmation linéaire, Problème de transport flou, Problème de transport à indices multiples, Algorithmes génétiques, Algorithmes hybrides, Métaheuristiques,

Abstract:

In this thesis, we are interested in solving the four-index fuzzy transportation problem by two methods, one is exact and the other is approximate. In the first part, we present an adaptation of an exact method introduced by Zitouni et al. in order to solve such a problem in the crisp case (as opposed to the fuzzy word).

Then, in the second part, we propose an approximate method (Metaheuristic). It is hybridization between Particle Swarm Optimization (PSO) and Genetic Algorithms (GA). It is essentially addressed to solve large problems where exact methods usually take a considerable amount of time to provide an optimal solution. At the end of this work, a comparative numerical study is given to test the robustness and efficiency of both approaches.

Keywords: Linear programming, Fuzzy transportation problem, Multi-index transportation problem, Genetic algorithms, Hybrid algorithms, Metaheuristics**.**