N° D'ORDRE : 29/2012-M/M.T

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE

HOUARI BOUMEDIENE

FACULTE DE MATHEMATIQUES

MEMOIRE

Présenté pour l'obtention du diplôme de MAGISTER

EN RECHERCHE OPERATIONNELLE

Spécialité : MATHEMATIQUES DE GESTION

Par AIT FEROUKH IDIR

Sujet

Techniques Métaheuristiques Multi-objectifs

Soutenu publiquement le 14 -12-2011 devant le jury composé de

Mr M.AIDER –Professeur à l'USTHB	Président
Mr D.CHAABANE–Professeur à l'USTHB	Directeur de Mémoire
Mr S.BOUROUBI –Professeur à l'USTHB	Examinateur
Mr M.MOULAI Professeur à l'USTHB	Examinateur

Remerciements

"Ô Seigneur ! A Toi la louange, une louange abondante, pure et bénie."

Je témoigne une reconnaissance particulière pour mon promoteur, que Mr. CHAABANE reçoive l'expression de toute ma reconnaissance pour ses précieuses contributions, conseils et orientations ainsi que ses aimables corrections et ses apports avantageux sur ce mémoire.

Mes plus sincères remerciements vont à : Mr.AIDER, Mr.BOUROUBI et Mr.MOULAI, membres du jury qui m'ont fait l'honneur d'en faire partie.

Finalement, je remercie toute personne ayant contribué de près ou de loin à l'élaboration de mon mémoire, sans oublier tous ceux qui m'ont encouragé le long de mon parcours universitaire.

Dédicaces

A ma mère, qui par sacrifice m'a mit au monde, qui par sacrifice a fait de moi ce que je suis…

A mon père, qui était et sera toujours mon école, à former des hommes…

Je sais que les paroles ne suffiront jamais à décrire ce que je ressens envers vous...mais je ne trouve pas mieux à dire que : " ***Ô mon Seigneur, fais-leur; à tous deux; miséricorde comme ils m'ont élevé tout petit".***

A mon petit frère et mes petites sœurs...

A la perle lumineuse qui éclaire ma vie…cadeau divin… Imène…

A tous ceux que j'aime…

Je dédie ce modeste travail…

Idir.

Sommaire

1 Introduction

Un problème d'optimisation se définit comme la recherche du minimum ou du maximum (de l'optimum donc) d'une fonction donnée. On peut aussi trouver des problèmes d'optimisation pour lesquels les variables de la fonction à optimiser sont contraintes d'évoluer dans une certaine partie de l'espace de recherche. Dans ce cas, on a une forme particulière de ce que l'on appelle un problème d'optimisation sous contraintes.

À partir de la fin de la Seconde Guerre mondiale, de nouvelles méthodes permirent de résoudre des problèmes complexes là où les méthodes classiques échouaient. Ces méthodes furent connues sous le nom de programmation linéaire, développées principalement par George B. Dantzig (né le 8 novembre 1914), mathématicien américain et créateur de la méthode du Simplexe, et L. Kantorovich (1912-1986). Danzig, outre la programmation linéaire, étudia entre autres la programmation mathématique, la prise de décision et les modèles de planification à large échelle. L'impact de son œuvre fut considérable en gestion et en économie et ses méthodes restent totalement d'actualité[8].

Les méthodes proposées jusque là reposèrent alors sur des modèles mathématiques à base d'un seul critère. L'avancée en matière à vu l'introduction de nouveaux modèles prenant en compte plusieurs critères conflictuels, supposés ainsi refléter mieux les problèmes concrets étudiés, ouvrant ainsi la porte à un nouveau type d'optimisation, en l'occurrence l'optimisation multi-objectifs.

Dès lors, un nombre abondant de chercheurs et de publications vinrent enrichir ce domaine demeurant toujours fertile, donnant naissance à de nouveaux horizons de mise en application de l'optimisation multi-objectifs, tels que la programmation linéaire multi-critères en variables entières, la programmation linéaire multi-critères en variables mixtes, la programmation fractionnaire linéaire multi-critères... etc.

Dans ce travail, un type particulier de problèmes est abordé, il s'agit des problèmes des flots maximaux multi-objectifs, ou une restriction existe sur la diminution de la valeur du flot de chaque arc. La valeur du flot passant par chaque arc est à maximiser, en même temps que la fonction objectif globale. Ce travail s'esquisse par un point de situation sur l'état de l'art de l'optimisation multi-objectifs, passant par un aperçu sur Programmation Multi-objectifs en nombres entiers. Il est exposé par la suite le problème des flots maximaux multi-objectifs ainsi que l'approche de résolution utilisée consistant à combiner la méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes et la recherche Tabou . Finalement, il est conclu par un exemple illustratif mettant en application l'algorithme de résolution adapté.

2 Optimisation Multiobjectif : Etat de l'art

2.1 Introduction

Habituellement, et d'une manière classique, la modélisation des problèmes de recherche opérationnelle conduit à un modèle mono critère sur un ensemble défini de contraintes. Ceci dit, de larges types de problèmes de planification et de décision impliquent souvent plusieurs objectifs conflictuels devant être considérés simultanément. Dans la programmation mathématique multi-objectifs, l'ensemble des solutions réalisables n'est pas connu explicitement à l'avance, mais délimité par un certain nombre de contraintes.

La recherche de solution ne générera plus une solution unique mais une multitude de solutions appelées solutions efficaces ou de Pareto et l'ensemble de solutions que l'on obtient à la fin de la recherche caractérise la surface de compromis.

2.2 Notations et Notions Fondamentales

Nous allons exposer dans ce qui suit quelques notions fondamentales de la programmation multi-objectifs.

Définition 1 *Un problème multi-objectifs ou multi-critères est le problème consistant à optimiser (maximiser ou minimiser) p fonctions objectifs (ou critères) simultanément avec* $p \geq 2$.

Mathématiquement, ce problème peut être formulé comme suit :

$$\begin{cases} Optimiser \quad [f_1(x), f_2(x), ..., f_p(x)] \\ x \in S \end{cases}$$

où :

$S = \{x \in \mathbb{R}^n \mid g_j(x) \leq 0,\ j = 1, 2, ..., m\}$;

f_k $(k = 1, ..., p)$ et g_j $(j = 1, ..., m)$ sont des fonctions à valeurs réelles du vecteur de décision $x \in \mathbb{R}^n$.

Si les objectifs (critères) f_k et les fonctions (contraintes) g_j sont linéaires en x , on obtient un problème de programmation linaire multi-objectifs ou MOLP pour désigner « **M**ultiple **O**bjective **L**inear **P**rogramming », s'écrivant sous la forme :

$$(MOLP) = \begin{cases} Optimiser \quad Z_k = c_k x,\ k = 1, ..., r \\ x \in S \end{cases}$$

$S = \{x = \mathbb{R}^n \mid Ax \leq b, x \geq 0\}; A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m; m, n \in \mathbb{N}$, $c_k \in \mathbb{R}^n$.

Sans perte de généralités, nous considérons dans ce qui suit que tout problème est à maximiser, le problème MOLP pourra être décrit comme suit :

$$(MOLP) = \begin{cases} \max \quad Z = Cx \\ x \in S \end{cases}$$

où C est une $p \times n$ matrice avec c^i comme vecteur ligne.

Remarque: *l'écriture "Optimiser" ou bien "maximiser" ou "minimiser" est controversée, ceci vient du fait que contrairement aux problèmes uni-critères, il n'existe pas en général un unique vecteur de valeurs qui soit meilleur que tous les autres.*

2.2.1 Concepts de base

Définition 2 (Espace des décisions) *L'espace $\mathbb{R}^n$ dans lequel se situe l'ensemble des actions S ($S \subseteq \mathbb{R}^n$) est appelé* ***Espace des décisions.*** *[22]*

Définition 3 (Espace des critères) *L'espace $\mathbb{R}^r$ dans lequel se situe Z_S est appelé* ***l'espace des critères :*** *Dans le cas linéaire, Z_S est l'image de S dans $\mathbb{R}^r$ par l'application linéaire associée à la matrice $C = (c_1, c_2, ..., c_r)$:*

$$Z_S = C(S) : x = (x_1, ..., x_n) \in S \rightarrow Cx = (Z_1, ..., Z_r) \in Z_S \quad [22]$$

Définition 4 (Dominance) *Soient deux vecteurs critères $Z, Z' \in Z_S$. On dit que Z* ***domine*** *Z' si et seulement si $Z \geq Z'$ et $Z \neq Z'$ (i.e $Z \geq Z'$ pour tout $i = 1, ..., r$, et $Z_i > Z'_i$ pour au moins un indice i).*

Si Z domine Z', alors Z est au moins aussi bon que Z' sur tous les critères et meilleur que lui sur au moins un critère. [22]

Définition 5 (Dominance forte) *Soient deux vecteurs critères $Z, Z' \in Z_S$. On dit que Z* ***domine fortement*** *Z' si et seulement si $Z > Z'$ (i.e $Z_i > Z'_i$ pour tout $i = 1, ..., r$).*

Si Z domine fortement Z', alors Z est meilleurs que Z' sur tous les critères. [22]

Définition 6 (Efficacité) *Une solution $x^* \in S$ est une solution efficace s'il n'existe pas de $x \in S$ telque $Z(x)$ domine $Z(x^*)$.*

Un point est efficace si son image par Z est un vecteur critère non dominé. Le terme efficacité est aussi connu sous **Pareto optimalité** ou **non infériorité.** Une définition équivalente de l'efficacité est :

Une solution $x^* \in S$ est une solution efficace s'il n'existe pas de $x \in S$ tel que

$$Z_i(x) \geq Z_i(x^*); i = 1, ..., r$$

avec au moins une inégalité stricte.

A partir d'un point efficace, il est impossible d'augmenter la valeur d'un des critères sans diminuer la valeur d'au moins un autre critère. [22]

Définition 7 *Une solution $x^* \in S$ est une solution **faiblement efficace** s'il n'existe pas de $x \in S$ telque $Z(x) > Z(x^*)$.*

Une solution est faiblement efficace si son vecteur critère n'est pas fortement dominé. Le terme faiblement efficace est connu aussi sous le nom de salter-optimale. [22]

Définition 8 (Efficacité forte) *Une solution $x^* \in S$ est une solution **fortement efficace** s'il n'existe pas de $x \in S$ tel que $x \neq x^*$ et $Z(x) \geq Z(x^*)$.*

Une solution x est fortement efficace s'il n'existe pas une autre solution telle que le vecteur critère, qui lui est associé, soit aussi bon que celui de x . Remarquons que l'efficacité forte implique l'efficacité qui implique à son tour l'efficacité faible.[22]

Définition 9 (Le point idéal) *On entend par* ***point idéal*** *le vecteur défini par :*

$$\overline{Z} = (\max_{x \in S} Z_1(x), ..., \max_{x \in S} Z_r(x)) \in \mathbb{R}^r$$

en général $\overline{Z} \notin Z_S$. [22]

Remarque *Les coordonnées de ce point sont obtenues en optimisant chaque fonction objectif séparément.*

Définition 10 (le point anti-idéal) *le vecteur défini par :*

$$\underline{Z} = (\min_{x \in S} Z_1(x), ..., \min_{x \in S} Z_r(x)) \in \mathbb{R}^r$$

constitue le point anti-idéal; en général $\underline{Z} \notin Z_S$. [22]

Définition 11 (Matrice des gains) *La matrice carrée de dimension r suivante :*

$$\begin{bmatrix} \overline{Z}_1 & Z_{12} & ... & Z_{1r} \\ Z_{21} & \overline{Z}_2 & ... & Z_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ Z_{r1} & Z_{r2} & ... & \overline{Z}_r \end{bmatrix}$$

où $\overline{Z}_i = \max_{x \in S} Z_i(x) = Z_i(\overline{x}_j), \quad \forall i, j = 1, ..., r$

avec $Z_{ij} = Z_i(\overline{x}_j), \ \forall i, j = 1, ..., r$

est appelée **la matrice des gains.** [22]

Définition 12 (Le point nadir) *Le point de R^r de coordonées :*

$$n_i = \min_{j=1,...,r} Z_{ij}, \ \ i = 1, ..., r$$

est appelé **le point nadir.** [22]

Remarque Les coordonnées de ce point correspondent aux pires valeurs obtenues par chaque fonction objectif lorsque l'on restreint l'espace des solutions à la surface de compromis.

Définition 13 (Face) *Le sous-ensemble non vide F du polyèdre S est une face s'il existe un polyèdre L tel que $F = L \cap S$ et S est tout entier situé dans un demi-espace fermé délimité par L mais non iclus à L. [5]*

Définition 14 (p-facette) *Soit F un sous-ensemble du plyèdre S et L un hyperplan de support de S . Alors F est dit une p-facette de S si et seulement si $L \cap S = F$ et F de dimension égale à p. Les poins extrêmes sont des faces de dimension* 0, *les arêtes de S sont des faces de dimension 1.[5]*

Définition 15 *Une face F de S est dite efficace si tout $x \in F$ est efficace.[5]*

2.3 Méthodes de résolution des problèmes multiobjectifs

2.3.1 Méthodes Exactes

Dans ce qui suit, nous allons décrire quelques méthodes de résolution des problèmes multi-objectifs. Nous nous limiterons à exposer seulement leur principe dans le but de s'épargner du fondement de l'approche de résolution. Il existe un nombre important de méthodes, classées en cinq groupes :

–les méthodes scalaires,

–les méthodes interactives,

–les méthodes floues,

–les méthodes exploitant une métaheuristique,

–les méthodes d'aide à la décision.

Les méthodes de ces cinq groupes peuvent aussi être rangées en trois familles de méthodes d'optimisation multi-objectifs [6] :

- **Les méthodes à préférence à priori** :

Dans ces méthodes, l'utilisateur définit le compromis qu'il désire réaliser (il fait part de ses préférences) avant de lancer la méthode d'optimisation.On retrouve dans cette famille la plupart des méthodes par agrégation (où les fonctions objectifs sont fusionnées en une seule).

Exemples des méthodes à préférence à priori :

Le principe des ces méthodes et de tenter de ramener le problème de programmation linéaire multi-Objectifs en un problème de programmation linéaire mono-objectif via des transformations reposant sur des fonctions scalarisantes telle la méthode qui consiste à remplacer les divers objectifs par une somme pondérée de la forme :

$$\max_{x\in S}\sum_{k=1}^{r}\lambda_k Z_k(x).$$

La somme pondérée n'est qu'un cas (le plus simple d'ailleurs) des fonctions scalarisantes, il existe néanmoins plusieurs fonctions dont les plus courantes sont :

- Caractérisation à l'aide de poids

$$S_1(Z,\lambda)=\sum_{k=1}^{r}\lambda_k Z_k(x) \qquad \text{avec } \sum\lambda_k=1, \lambda_k>0, \forall k\in\{1,...,r\}.$$

$$S_2(Z,\lambda)=\sum_{k=1}^{r}\lambda_k\left|Z_k-\overline{Z}_k\right|.$$

- Norme L_r pondérée :

$$S_3(Z,\lambda)=\left[\sum_{k=1}^{r}\lambda_k\left|Z_k-\overline{Z}_k\right|^r\right]^{\frac{1}{r}}, r\in Z_0^+.$$

- Norme L_r de Tchebytcheff pondérée :

$$S_4(Z,\lambda)=\max_{1\le k\le r}\left\{\lambda_k\left|Z_k-\overline{Z}_k\right|\right\}.$$

- Caractérisation à l'aide de poids cibles

- Niveaux d'aspiration :

$$S_5(Z,\lambda) = \left[\sum_{k=1}^{r}\lambda_k \left|Z_k - \widehat{Z}_k\right|^r\right]^{\frac{1}{r}}, \; r \in Z_0^+ \text{ et } \widehat{Z} \in Z_S.$$

- Niveaux de résérvation :

$$S_6(Z,\lambda) = \left[\sum_{k=1}^{r}\lambda_k \left|Z_k - \widehat{Z}_k\right|^r\right]^{\frac{1}{r}} \text{ est des contraintes sur les critères}$$

$$Z_k \geq \widehat{Z}_k, \qquad \forall k \in \{1,...,r\} \; .[22]$$

- **Les méthodes à préférence progressive :**

Les méthodes appartenant à cette famille reposent sur un principe qui consiste à faire intervenir le décideur au fur et a mesure du déroulement de l'algorithme dans le but d'affiner le choix de compromis. Il est question alors d'une alternance permanente entre le calcul algorithmique et l'ajustement des compromis de la part du décideur en apportant des informations complémentaires sur leurs préférences. Celles-ci sont directement injectées dans l'algorithme afin de fournir une nouvelle base de construction de nouveaux compromis.

Exemples des méthodes à préférence progressive :

Step Method où Méthode des étapes :

Le prinipe de cette méthode consite à réduire progressivement le domaine $\psi(S)$ par l'addition de contraites sur les valeurs critères. Un nouveau compromis est généré à chaque étape en optimisant une norme de type :

$$S(Z,\lambda) = \max_{1\leq k\leq r}\left\{\lambda_k \left|Z_k - \overline{Z}_k\right|\right\} + \rho\sum_{k=1}^{r}\lambda_k \left|Z_k - \overline{Z}_k\right|; \rho > 0,$$

sur une partie de $\psi(S)$ selon une direction correspondante à la relaxation d'un critère fixé par les décideurs.

Cette méthode passe par les autres étapes suivantes :

Etape 1 : Déterminer la matrice des gains ainsi que les coefficients normalisés suivants :

$$\lambda_j = \frac{\alpha_j}{\sum\limits_i \alpha i} \text{ où } \alpha_j = \frac{\overline{Z} - n_j}{\max\left\{\left|\overline{Z}\right|, |n_j|\right\}} \text{ pour } j = 1, ..., r;$$
$$\eta = (n_j)_{j=1,...,r;} \text{ est le point nadir .}$$
$$\text{Soit } \psi(S^1) \text{ et } k = 1.$$

Etape 2: Calculer la solution de compromis Z^k en résolvant le problème :

$$\begin{cases} \min\left(\mu\sum\limits_{j=1}^{r}\rho_j Z_j\right) \\ t.q \\ \mu \geq \lambda_j\left(\overline{\overline{Z_j}} - Z_j\right) \text{ pour } j = 1, ..., r \end{cases}$$

avec $\overline{\overline{Z_j}} = Z_j + \varepsilon_j$ et ε_j, ρ_j sont positifs très petits. $\overline{Z_j}$ est le $j^{ème}$ élément de la diagolane principale de la matrice des gains.

Etape 3: Présenter la solution Z^k obtenue au décideur.

- Si Z^k satisfait le décideur alors fin.

- Sinon, lui demander d'indiquer sur quel critère (indice) i est-il prêt à faire une concession et quelle quantité maximum Δ_i accepte t-il de concéder.

Etape 4: Réduire l'ensemble des résultats potentiels à :

$$\psi(S^{k+1}) = \left\{ Z \in \psi(S^k) \mid Z_i \geq Z_i^k - \Delta_i \text{ et } Z_j \geq Z_j^k \ \ \forall j \neq i \right\}$$

faire $\lambda_i = 0, k = k + 1$ et aller à l'étape1. [4] [5] [6]

- **Les méthodes à préférence à posteriori** :

Dans cette catégorie, un groupe de solutions mathématiquement équivalents est présenté au décideur après le déroulement de l'algorithme de résolution, afin d'en choisir un ou plusieurs compromis qu'il estime les plus satisfaisants. Il est à souligner que dans cette approche, il n'est pas nécessaire de disposer d'une connaissance au préalable sur la nature ou la structure des préférences du décideur.

2.3.2 Méthodes exploitant une Métaheuristique

La résolution des problèmes concrets du monde réel demeure toujours un véritable challenge, les chercheurs font souvent face à des problèmes n'ayant pas jusque là déjà été modélisé. Les contraintes de taille et de complexité algorithmique rendent l'utilisation des méthodes exactes extrêmement difficile, c'est pour cela que les recherches tendent de plus en plus vers les métaheuristiques.

Le recours aux métaheuristiques dans l'optimisation multi-objectifs intervient comme conséquence logique du succès des ces dernières dans l'optimisation mono-objectif, et ne cesse de être un des domaines de recherche ayant une croissance des plus rapides.

Nous distinguons le recours particulier aux algorithmes évolutionnaires, donnant naissance à l'Optimisation Evolutionnaire Multi-Objectifs ou « EMO » pour «Evolutionary Multiobjective Optimization ». Le principe de base de l'algorithme évolutionnaire demeure le même, à savoir l'initialisation d'une population, la sélection, le crossover, la mutation, l'élitisme, la création de la nouvelle population et le test final, cependant la population représentera dans le cas multi-objectifs une ou l'ensemble des solutions réalisables dans la perspective de les améliorer afin d'aboutir à un ensemble de solutions Pareto-Optimale.

Il existe de nombreuses méthodes basées sur les Algorithme Evolutionnaires : Non-dominated Sorting Genetic Algorithm , Niched-Pareto Genetic Algorithm, Multiobjective Genetic Algorithm,... etc. la plus part de ces méthodes sont détaillé les références [4] [9] [25] et surtout [1] .

En addition, dés méthodes Métaheuristiques multi-objectifs basées sur la recherche locale émergent toujours, l'algorithme de Recherche des Sommets Adjacents (Adjacent Vertex Search Algorithm) proposé par Yammamoto [28] en est un exemple, d'autant plus celles basée sur la Recherche Tabou [9].

3 Programmation Multi-objectifs en nombres entiers

La modélisation des problèmes d'optimisation multi-objectifs a permis de résoudre une large gamme de cas réels et concrets. Ceci dit, il existe un nombre considérable de cas ou les variables de décision prennent des valeurs entières. En effet, faire face à des problèmes de planification, de plans de production, où de plus à des problèmes des flots multi-objectifs rend inéluctable le passage par la programmation multi-objectifs en nombres entiers.

3.1 Introduction

Souvent dans la théorie des flots dans les réseaux, on l'en considère que chaque flot passant par chaque arc est totalement contrôlable d'une manière ou on peut augmenter ou diminuer le flot d'un arc donné. Dans ce qui suit on considère que la réduction d'un flot passant par un arc donné n'est pas permise. Le problème sera formulé sous forme d'une optimisation d'un problème multi-critères sur un ensemble de solutions efficaces.

Lorsque toutes les variables de décision d'un quelconque programme linéaire sont entières, il s'agit alors d'un programme linéaire en variable entières ou discrètes, tandis que si seulement certaines d'entres elles sont entières, il s'agit alors d'un programme linéaire en variables mixtes.

3.2 Formulation des Problèmes mono-critères en nombres entiers

Soit la formulation suivante du problème de programmation linaire mono-critère :

$$\begin{cases} \max & Z = cx \\ t.q & x \in S \end{cases}$$

avec $S = \{x = \mathbb{R}^n \mid Ax = b, x \geq 0\}; A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m; c \in \mathbb{R}^{1 \times n}$

La formulation mathémathique du problème de programmation linéaire en variables mixtes s'écrit alors [5] :

$$\begin{cases} \max & (cx + hy) \\ t.q & Ax + Gy = b \\ & x \geq 0 \\ & y \in \mathbb{N} \end{cases}$$

La formulation mathémathique du problème de programmation linéaire en variables discrètes est donnée par :

MOILP (**M**ulti **O**bjective **I**nteger **L**inear **P**rogramming) $\begin{cases} \max & Z = cx \\ t.q & x \in D \end{cases}$

où $D = S \cap \mathbb{Z}^n, \mathbb{Z}$ est l'ensemble des nombres entiers relatifs.

ou bien

$$\begin{cases} \max\limits_{x \in D} & Z_k(x) = c_k x, k = 1, ..., r \\ t.q & D = \{x \in S; \quad x \text{ entier}\} \end{cases}$$

3.3 Solutions efficaces supportées

La nature discrète des variables de décision du problème MOILP engendrent un ensemble d'admissibilité ayant une structure en points, autrement dit l'ensemble d'admissibilité n'est pas convexe.

Soit le problème paramétrique :

$$S_1(Z, \lambda) = \sum_{k=1}^{p} \lambda_k Z_k(x) \qquad \text{avec } \sum \lambda_k = 1, \lambda_k > 0, \forall k \in \{1, ..., r\}.$$

Nous appellerons ensemble des solutions efficaces supportées, dénoté par X_{ES} l'ensemble des solutions efficaces générées par le problème :

$$\max_{x \in D} S_1(Z, \lambda).$$

3.4 Solutions efficaces non-supportées

Soit X_E l'ensemble des solutions efficaces du problème MOILP, nous appellerons alors ensemble des solutions efficaces non-supportées, dénoté par X_{ENS} l'ensemble :

$$X_{ENS} = X_E \setminus X_{ES} .$$

4 Optimisation sur l'ensemble des solutions efficaces

4.1 Introduction

Les études et recherches portant sur l'optimisation d'une fonction sur l'ensemble des solutions efficaces ont connue une évolution substantielle depuis que ce type de problèmes à été soulevé pour la première fois par Philip en 1972[23]. L'optimisation d'une fonction sur l'ensemble des solutions efficaces est la maximisation d'une fonction ϕ donnée sur l'ensemble des points x de l'ensemble des solutions réalisables X de R^n tel qu'il n'existe aucun autre point x' satisfaisant $(c^1x', ..., c^rx') \geq (c^1x, ..., c^rx)$ et $(c^1x', ..., c^rx') \neq (c^1x, ..., c^rx)$ où $c^1x, ..., c^rx$ est un r vecteur critère. La difficulté capitale caractérisant ce genre de problème réside dans la non-convexité d'un tel ensemble de recherche.

Considérons le problème d'optimisation sur l'ensemble des solutions efficaces suivant :

$$(MC) \begin{cases} \max & Cx \\ t.q & x \in X \end{cases}$$

où C est une $p \times n$ matrice ayant comme lignes les vecteures c^i, et X est un ensemble polyèdrale de R^n définit comme suit : $X = \{x \mid x \in R^n; Ax =$

$b; x \geq 0\}$. Soit X_E l'ensemble des points efficaces, le problème est alors formulé comme suit :

$$(P_E) \begin{cases} \max & \phi(x) \\ t.q & x \in X_E \end{cases}$$

où $\phi : R^n \longrightarrow R$ est une fonction continue à maximiser. Comme déjà cité, la difficulté de ce type de problèmes provient principalement de la non-convexité de l'ensemble X_E. [28]

4.2 Définitions

Définition 16 *Un point $x \in R^n$ est appelé point efficace du problème (MC) si $x \in X$ et il n'existe aucun point x' tel que $Cx \leq Cx'$ et $Cx \neq Cx'$. L'ensemble des points efficaces du problème (MC) est dénoté par X_E.Un point $x \in R^n$ est appelé point faiblement efficace du problème (MC) si $x \in X$ et il n'existe aucun point x' tel que $Cx < Cx'$. L'ensemble des points faiblement efficaces du problème (MC) est dénoté par X_W.[28]*

Définition 17 *L'ensemble $Y = CX = \{y \mid y \in R^r; y = Cx \text{ pour } x \in X\}$ est appelé l'ensemble des résultats. L'ensemble $Y^{\leq} = Y \mid R^r_- = \{y \mid y \in R^r; y \leq Cx \text{ pour } x \in X\}$ est appelé l'ensemble des résultats inférieurs et $Y^{<} = Y \mid R^r_- \_ = \{y \mid y \in R^r; y < Cx \text{ pour } x \in X\}$ appelé l'ensemble des résultats strictement inférieurs. [28]*

Définition 18 *Un point $y \in Y$ est appelé un résultat efficace (Solution non dominée) s'il n'existe aucun point $y' \in Y$ tel que $y \leq y'$ et $y \neq y'$, en d'autres termes $Y \cap (y \mid R^r_+) = \{y\}$. On note l'ensemble des solutions non dominées par Y_E. Un point $y \in Y$ est dit faiblement non dominé si $Y \cap (y + R^r_{++}) = \emptyset$. Il est dénoté par Y_W.[28]*

Définition 19 *pour $\lambda \in R^r_{++}$ et $x \in X$ soit la fonction*

$$g_\lambda(x) = \max\{\lambda Cx' \mid x' \in X; Cx' \geq Cx\} - \lambda Cx$$

est appelée la fonction gap. La fonction gap est dénotée g dans le cas où $\lambda = e = (1, ..., 1)$. [28]

Théorème 20 $X_E = \{x \mid x \in X; \exists \lambda \in R^r_{++}$ *tel que* $\lambda Cx' \geq \lambda Cx \ \forall x' \in X\}$.

$$= \{x \mid x \in X; \nexists x' \in R^n \text{ tel que } Cx' \geq 0;\ Cx' \neq 0;$$

$Ax\prime = 0; x'_i \geq 0$ pour i avec $x_i = 0\}$.

$$= \{x \mid x \in X; \exists(\lambda, \mu, \nu) \in R_{r++} \times R_m \times R_{n+} \text{tel que}$$

$\lambda C - \mu A + \nu = 0; \nu x = 0\}$.

$$= \{x \mid x \in X; g_\lambda(x) = 0\}\ .\quad [28]$$

Théorème 21 *L'ensemble X_E est connexe. Deux sommets quelconques sont reliés par un chemin d'arêtes efficaces. [28]*

4.3 Algorithme de Recherche des Sommets Adjacents (Adjacent Vertex Search Algorithm)

Y.Yammamoto [28] propose une méthode d'optimisation sur l'ensemble des solutions efficaces se basant sur un algoritme de recherche des sommets adjacents :

Algorithme 22 *Algorithme de Recherche des Sommets Adjacents (Adjacent Vertex Search Algorithm)*

[0] *(Initialisation)*

Soit $p = k = 0, X^0 = X$ *et trouver* $x^0 \in X_v \cap X_E$.

Si $N_E(x^0) = \emptyset$ *alors* x^0*est une solution optimale de* (P_E). *Sinon, aller à l'étape [1]*

[1] *(Boucle majeure)*

si $\{x \mid x \in N_E(x^p); \phi(x) > \phi(x^p)\} \neq \emptyset$,*choisir* x^{p+1} *de cet ensemble, mettre* $p = p + 1$ *aller à l'étape [1]*

sinon, soit $L^p = \{x \mid \phi(x) \leq \phi(x^p)\}$,*aller à l'étape [2]*

[2] *(Boucle mineur)*

2.1 *trouver* $\nu^k \in \arg\max\{\phi(x) \mid x \in X^k\}$. *Si* $\phi(x^p) \geq \phi(\nu^k) - \epsilon$ *pour une tolérance* $\epsilon > 0$,*alors arreter avec* x^p *comme une* $\epsilon-$*approximation de la solution optimale. Sinon aller à (2.2) .*

2.2 *Trouver un Hyperplan suppot* H^k *de* L^p *tel que* $L^p \subseteq H^k_+$ *et* $\nu^k \in H^k_-$ _

2.3 *S'il existe une arrête efficaces* $[\mu', \mu'']$ *tel que* $[\mu', \mu''] \cap H^k \neq \emptyset$ *et* $\max\{\phi(\mu'), \phi(\mu'')\} > \phi(x^p)$,*alors mettre* x^{p+1}*l'un des* μ' *et* μ'' *ayant la valeur*

la plus grande valeur de la fonction objectif. Mettre $p = p + 1$ *et aller à l'étape [1].*

Sinon aller à (2.4).

2.4 *Mettre* $X^{k+1} = X^k \cap X^k_+$, $k = k + 1$ *et aller à l'étape [2].*

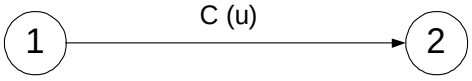
5 Problème des flots maximaux

5.1 Caractérisation du problème du flot maximal

Dans cette section nous allons estomper une caractérisation du problème du flot maximal, nous allons également faire le contraste entre un flot maximal et un flot maximum.

5.1.1 Problèmes de flots :

Définition 23 (Réseau de transport) *Un réseau de transport est un graphe sans boucle ou à chaque arc (u) est associé un nombre $c(u)$ appelé «capacité» de l'arc u.*



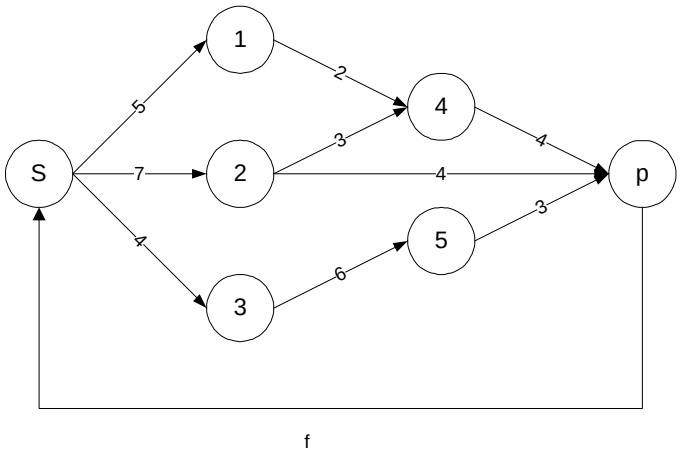
$c(u)$: Capacité de l'arc u reliant les sommets 1 et 2.

Ainsi un réseau de transport R est défini comme suit :

$R = (X, U, c)$ où $G = (X, U)$ est un graphe orienté, $|U| = m$.

La capacité d'un arc est définie par $c : U \to R^+ \cup \{+\infty\}$

Définition 24 (Flot) *Un flot dans un réseau est une fonction qui associe à chaque arc (u) une quantité $f(u)$ qui représente la quantité de flot qui y passe, en provenant d'une source s, et en destination d"un puits p.*



Par conséquence, le problème du flot maximal consiste à trouver la valeur maximale du flot (f) tout en respectant les contraintes suivantes :

Définition 25 (Contraintes de capacité) *Etant donné qu'a chaque arc (u) est associée une capacité $c(u)$, le problème de recherche du flot maximal doit tenir compte que la solution donnée sur chaque arc ne dépasse point sa capacité. Ainsi, pour tout arc (u), la quantité de flot qui y passe $f(u)$ doit être inférieure ou égale à sa capacité $c(u)$. Nous avons donc :*

$$0 \leq x_h \leq c_h \forall h \in U$$

Définition 26 (Contraintes de conservation) *Le flot circulant sur le graphe doit respecter la loi de Kirchhoff, la quantité du flot entrant doit être égale à la quantité du flot sortant pour chaque sommet :*

$$\sum_{\partial^+ h=i} x_h = \sum_{\partial^- h=i} x_h \forall i \in V \setminus \{s,t\}$$

5.1.2 Formulation du problème du flot max

Bien que le problème du flot maximum est souvent formulé et modélisé moyennant les graphes dans l'objectif de parvenir à une résolution par l'algorithme de Ford-Fulkerson, nous allons procéder dans ce qui suit à une formulation en programmation linéaire.

Le réseau des flots sera caractérisé par des notations suivantes :

V : L'ensemble des sommets (nœuds) du réseau.

U : L'ensemble des arcs du réseau.

s : Le sommet source.

t : Le sommet d'arrivée.

$\partial^+ h$: L'extrémité initiale de l'arc h.

$\partial^- h$: L'extrémité finale de l'arc h.

c_h: Capacité de l'arc h.

Le vecteur$x = (\ldots, x_h, ...)$ de dimension $|U|$ est appelé flot réalisable s'il satisfait les contraintes de conservation et de capacités suivantes :

Contraintes de conservation :

$$\sum_{\partial^+ h=i} x_h = \sum_{\partial^- h=i} x_h, \forall i \in V \setminus \{s,t\}\,.$$

Contraintes de capacité:

$$0 \leq x_h \leq c_h \forall h \in U.$$

On définit la matrice d'incidence sommets arcs de la manière suivante :

$$a_{ih} = \begin{cases} +1 & si \quad \partial^+ h = i \\ -1 & si \quad \partial^- h = i \\ 0 & sinon \end{cases}$$

Ce qui permet d'écrire l'équation de conservation d'une manière plus simple :

$$Ax = 0.$$

Un flot réalisable x est appelé un flot maximal s'il n'existe pas un autre flot réalisable $x\prime$ tel que $x\prime \geq x$ et $x\prime \neq x$.

L'objectif est de maximiser la valeur globale du flot, ce qui implique la maximisation de la somme des valeurs des flots passant par chaque arc, la fonction objectif s'écrit alors sous la forme :

$$Z_{\max} = \sum_{h=1}^{|U|} x_h.$$

Ainsi, le programme linéaire associé au problème du flot maximum s'écrit sous la forme :

$$(PL) = \begin{cases} \sum_{\partial^+ h=i} x_h = \sum_{\partial^- h=i} x_h \ , & \forall i \in V \setminus \{s,t\} \\ 0 \leq x_h \leq c_h \ , & \forall h \in U \\ Z_{\max} = \sum_{h=1}^{|U|} x_h \\ x \geq 0 \end{cases}$$

5.1.3 Problème du flot maximal

Dans ce qui précède, le problème du flot maximum consiste à maximiser la valeur du flot global traversant le réseau de transport tout en supposant que le flot de chaque arc est totalement contrôlable, autrement dit, nous nous permettons d'augmenter ou de diminuer la valeur tant que les conditions de conservation et de capacité sont respectées.

Dans le problème du flot maximal, une restriction existe sur la diminution de la valeur du flot de chaque arc. La valeur du flot passant par chaque arc est à maximiser, en même temps que la fonction objectif globale, ou bien en définissant une fonction objectif à optimiser (à maximiser ou à minimiser) sur l'ensemble des solutions trouvées à partir de ce problème. Nous parlons alors de la détermination des flots maximaux. Il est à souligner que le problème

des flots maximaux appartient à la classe NP-difficile, la difficulté principale réside dans le fait que le problème engendre autant de fonctions objectifs que le nombre d'arêtes dans le graphe, ainsi un simple graphe à cinq (05) arêtes par exemple engendre un problème à cinq (05) objectifs.

5.1.4 Formulation du problème des flots maximaux

Etant donné un réseau $G = (V, s, t, \partial^+ h, \partial^- h, c_h)$, le problème des flots maximaux consiste à maximiser le flot passant par chaque arc, ainsi il en ressort un problème d'optimisation multi-critères en nombres entiesr ou le vecteur des fonctions objectifs à optimiser est donné par :

$$MaxZ_k(x) = x_k \text{ avec } k = 1..|U|$$

Le problème s'écrit alors sous la forme :

$$PLE = \begin{cases} \sum_{\partial^+ h=i} x_h = \sum_{\partial^- h=i} x_h \,, & \forall i \in V \setminus \{s,t\} \\ 0 \leq x_h \leq c_h, & \forall h \in U \\ MaxZ_k(x) = x_k, & k = 1..|U| \\ x \in N \end{cases}$$

Les contraintes de conservations peuvent s'écrire sous la forme $Ax = 0$ ou A est une matrice de dimension $|V \setminus \{s,t\}| \times |U|$ dont les éléments sont notés par

$$a_{ih} = \begin{cases} +1 & si \quad \partial^+ h = i \\ -1 & si \quad \partial^- h = i \quad ; i \in V \setminus \{s,t\} \, ; h \in U \\ 0 & sinon \end{cases}$$

La matrice A représente en conséquence la matrice d'incidence sommets-arcs dont les éléments sont dénotés par a_{ih} ou :

$$a_{ih} = \begin{cases} +1 & \text{si le sommet } i \text{ est l'extremité initiale de l'arc } h \\ -1 & \text{si le sommet } i \text{ est l'extremité finale de l'arc } h \\ 0 & sinon \end{cases}$$

5.1.5 Résolution des Problèmes des flots maximaux par optimisation sur l'ensemble des solutions efficaces

L'approche de résolution d'un problème de flot maximal consiste à considérer le problème cité dans le chapitre précédent (PLE) sous la forme :

$$(MC) \quad \left| \begin{array}{ll} \text{vecteur à maximiser} & Cx \\ \text{sujet à} & x \in X \end{array} \right.$$

(MC) caractérise donc un programme linéaire Multi-objectifs en variables discrètes.

Nous admettons que l'ensemble X est borné, nous décrivons X_v et X_E comme dénotations de l'ensemble des points extrêmes et des points efficaces du problème (MC) respectivement.

Pour rappel, un point $x \in R^n$ est dit *un point efficace* du problème (MC) si $x \in X$ et il n'existe aucun point $x' \in X$ tel que :

$$Cx' \geq Cx \text{ et } Cx' \neq Cx$$

Par conséquent, le problème d'optimisation sur l'ensemble des solutions efficaces relatif au problème des flots maximaux sera décrit comme suit :

$$(P) \quad \left| \begin{array}{ll} \text{optimiser} & f(x) \\ \text{sujet à} & x \in X_E \end{array} \right.$$

où $f(x)$ est une fonction à optimiser.

Principe : L'approche de résolution se fera en deux phases majeures :

- La résolution du problème (MC) en utilisant la Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes «SEEVD».
- L'optimisation d'une fonction objectif donnée sur l'ensemble des solutions efficaces engendré par la première phase moyennant une métaheuristique, à savoir la recherche tabou.

Nous allons procéder dans ce qui suit à la présentation explicite des deux méthodes utilisées. Il est à souligner que La résolution du problème (MC) se fera à travers la Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes « SEEVD » Initiée en 2007 par Mr D.Chaabane [5], Nous nous contenterons de reprendre ici la présentation générale de la méthode ainsi que de l'algorithme, Nous laisserons au lecteur le soin de recourir à la publication d'origine pour plus d'explications ou d'éventuels exemples d'application.

5.1.6 Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes « SEEVD »

La Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes «SEEVD» détermine toutes les solutions efficaces du problème (MOILP) sans n'en manquer aucune. Une solution initiale est déterminée en une première étape en résolvant le problème mono-objectif en optimisant les contraintes du problème (MC) sur un seul des critères $(PL1)$, puis une séquence de coupes $\sum_{j \in N_k \setminus \{j_k\}} x_j \geq 1$ et de Gomory sont appliquées après avoir exploré l'arête E_k. La méthode est une forme modifiée de la méthode de Gupta et Malhotra et le test d'arrêt est modifié pour produire toutes les solutions efficaces de la région d'admissibilité D. La technique est présentée en détail dans la section suivante.

Description de la méthode

• La méthode repose sur les techniques d'optimisation monocritère en nombres entiers. Les méthodes dual simplexe et les coupes de Gomory s'imposent vu l'architecture de l'algorithme.

Au départ, une solution optimale optimisant un des critères est déterminée.

• Dans le cas où cette solution est unique, le vecteur des critères correspondant est non dominé; cette solution constitue à elle seule la liste des solutions efficaces initiales. On explore par la suite les arêtes incidentes à cette solution selon une direction précise définie ultérieurement par un ensemble d'indices hors-base Γ.

Suivant chacune des directions, on calcule les solutions réalisables entières si elles existent, on évalue les critères sur chacune des solutions, les vecteurs critères correspondants qui sont dominés sont éliminés par une comparaison deux à deux et seuls les vecteurs critères non dominés sont retenus dans la nouvelle liste (la liste qui met à jour l'ancienne).

Lorsqu'une arête est sélectionnée, pour éliminer et réduire le domaine de recherche des solutions efficaces, une coupe de type ("Dantzig généralisée") est appliquée.

La sélection de l'arête peut se faire arbitrairement ou on choisit celle qui contient le plus de solutions réalisables . Ces deux options de choix ont

été testées et on a remarqué que sur quelques instances, le dernier choix peut causer une croissance du nombre d'itérations. Dans cette étape les méthodes dual simplexe et Gomory sont appliquées pour obtenir de nouveau une solution optimale, sur laquelle on évalue le vecteur critère; on l'ajoute 'à la liste précédente s'il n'est pas dominé. Le processus continue ainsi jusqu'à ce que le domaine réduit devienne vide.

Dans le cas où la solution initiale n'est pas unique, on cherche toutes les solutions optimales sur les arêtes incidentes dans les directions où les coûts réduits sont nuls. Sur chacune de ces solutions on évalue les vecteurs critères et on les compare deux à deux; seuls les vecteurs critères non dominés sont sauvegardés dans la liste des vecteurs non dominés. Les étapes citées précédemment dans la première phase sont effectuées (i.e. explorer les arêtes incidentes à la solution initiale, évaluer les vecteurs critères, mettre à jour la liste initiale et réduire le domaine de recherche des solutions efficaces) jusqu'à ce que le domaine réduit devienne vide . La liste finale contient toutes les solutions efficaces [5].

Voici une présentation plus formelle de l'algorithme.

Algorithme 27

Etape 1: Résoudre le problème (PL_1).une solution X_1^*dont les évaluations sont $(Z_1^1, Z_2^1, ..Z_p^1)$. Il est possible de résoudre un des problèmes $(PL_i; i = 2, 3, ..., p)$ maximisant $Z_i = c^i x$.

- Si $J_1 = \{j \in N_1 \mid Z_j^1 - c_j^1 = 0\} = \varnothing$, alors la solution optimale est unique,enregistrer le premier vecteur non dominé $(Z_1^1, Z_2^1, ..Z_p^1)$ et former la liste initiale des solutions non dominées. $Opt_1 = \{(Z_1^1, Z_2^1, ..Z_p^1)\}$. Aller à l'étape 2.

- Si $J_1 = \{j \in N_1 \mid Z_j^1 - c_j^1 = 0\} = \varnothing$, la solution optimale du problème (PL_1) peut ne pas être unique.Pour chaque $j \in J_1$ calculer $\Phi_j = \min_{i \in I_1}\{\frac{x_{1,i}}{y_{1,ij}}; y_{1,ij} \succ 0\}$

Si pour tout $j \in J_1$ on a $\Phi_j < 1$,il n'y a pas de solutions alternatives à la solution X_1 ; initialiser la liste des vecteurs potentiellement non dominés $Opt_1 = \{(Z_1^1, Z_2^1, ..Z_p^1)\}$ et aller à l"étape 2.

(b) Sinon,tant qu'il existe au moins un $j \in J1$ telque $\Phi_j \geq 1$ faire :

Exploiter l'arête

$$E_j = \left\{ (X_1') \in R^{(|I_1|+|N_1|)} \mid \begin{array}{lll} x'_{1,i} = x_{1,i} - \theta \times y_{1,ij} & & i \in I_1 \\ x'_{1,i} = \theta & & \\ x'_{1,\alpha} = 0 & & \text{pour tout } \alpha \in N_1 \backslash \{j\} \end{array} \right\}$$

pour θ entier variant entre 1 et Φ_j et $\theta \times y_{1,ij}$ entiers.

• Sur chacune des solutions réalisables trouvées sur une arête,évaluer les critères,et ajouter à la liste les vecteurs critères non dominés (une comparaison deux à deux est effectuée).

• Choisir arbitrairement un $j_1 \in J_1$ et aller à l"étape 2.2 où l'on réduira ledomaine à l'aide d'une coupe de type I. (On peut alternativement choisir j_1 tel que Φ_{j1} soit maximal).

Etape 2 : : Soit $k = 1$

2.1 Construire l'ensemble $\Gamma_k = \{j \in N_k \mid Z_j^1 - c_j^1 > 0$ et $\exists i \in \{2, ..., p\}$ tel que $Z_j^1 - c_j^1 > 0\}$.

2.1.1 si $\Gamma_k = \varnothing$ aller à l'étape 2.2 (la coupe devient une coupe de Dantzig $\sum_{j \in N_k} x_j \geq 1$)

2.1.2 Sinon,soit $\gamma = \Gamma_k$,et aller à (a).

(a) Si $\gamma = \varnothing$, choisir un $j_k \in \Gamma_k$ et aller à l'étape 2.2 . Sinon, soit $j_k \in \gamma$; calculer $\theta_{k,jk}^0 = 0$, il n'y a aucune solution réalisable sur l'arête E_{jk} ,faire $\gamma := \gamma \backslash \{j_k\}$ et aller à (a).sinon aller à (2.1.3)

2.1.3 Pour $\theta = 1, 2, ..., \theta_{k,jk}^0$; calculer toute les solutions réalisables entières sur l'arête E_{jk} en utilisant :

$$\begin{cases} x'_{1,i} = x_{1,i} - \theta \times y_{1,ij} \\ x'_{1,i} = \theta \\ x'_{1,\alpha} = 0 \end{cases}$$

Evaluer les critères sur chaque solution,éliminer les vecteurs critères dominés et introduire les nouveaux vecteurs critères potentiellement non dominés (non dominés par rapport aux solutions sur la liste à l'itération k) dans la liste Opt_k.Choisir un indice $j_k$de Γ_k et aller à l"étape (2.2).

2.2 Ajouter la contrainte $\sum_{j \in N_k \backslash \{j_k\}} x_j \geq 1$et appliquer la méthode du dual simplexe et Gomory si nécessaire. Soit X_{k+1} une solution optimale du problème augmenté. Evaluer tous les critères en cette

solution.Si le vecteur critère correspondant est dominé par un des éléments de la liste Op_{tk}, l'ignorer .Sinon,l'ajouter à la liste des vecteurs non dominés pour produire la liste Opt_{k+1}. Faire $k := k + 1$et aller à (2.1).

Etape 3 : La procédure prend fin quand l'opération pivot est impossible-le problème est devenu irréalisable dans la nouvelle région tronquée-l'algorithme est erminé et la liste finale Opt_{k+1} représente l'ensemble de toutes les solutions non dominées [5].

5.2 La Recherche Tabou

La Recherche Tabou est une méthaheuristique qui a été proposée par Fred Glover et Manuel Laguna en 1986 [14]. Depuis, la méthode est devenue très populaire grâce aux succès qu'elle a remportés en résolvant de nombreux problèmes.

La recherche tabou examine les solutions voisines de la solution courante, mais le mouvement déterminant la prochaine solution est fait vers la solution voisine la plus proche. Pour éviter un comportement cyclique de la méthode, les solutions qui ont été récemment examinées sont classées tabou, c'est-à-dire interdites pendant un certain nombre d'itérations. La recherche tabou peut en outre être améliorée par certains mécanismes, dont nous parlerons plus

loin dans ce chapitre. Les méthodes Recherche Tabou peuvent être réparties en deux catégories :

-Algorithme tabou de base : mémoire à court terme (liste taboue).

-Algorithme tabou évolué : mémoire à court terme (liste tabou) + mémoire à long terme pour assurer l'intensification et/ou la diversification.

5.2.1 Concepts de base

L'idée de base de la liste taboue consiste à mémoriser les configurations ou régions visitées et à introduire des mécanismes permettant d'interdire à la recherche de retourner trop rapidement vers ces configurations. Ces mécanismes sont des interdictions temporaires de certains mouvements (mouvements tabous). Il s'agit d'interdire les mouvements qui risqueraient d'annuler l'effet de mouvements effectués récemment.

A chaque itération, l'algorithme tabou choisit le meilleur voisin non tabou, autrement dit le meilleur voisin non contenu dans la liste des restrictions, même si celui-ci dégrade la fonction de coût. Pour cette raison, on dit de la recherche avec tabou qu'elle est une méthode agressive.

Ainsi, le principe général de la recherche tabou peut être énoncé de la manière suivante : étant donné une fonction f à optimiser sur l'ensemble X, la méthode de recherche tabou construit une solution initiale, puis procède itérativement d'une solution courante à une nouvelle solution courante, jusqu'à ce qu'un critère d'arrêt soit satisfait. À chaque solution $x \in X$ est associé un

voisinage $V(x) \in X$ des nouvelles solutions potentielles, et chaque solution $x\prime \in V(X)$ est atteinte à partir de x par une opération appelée mouvement.

La recherche tabou dépasse un optimum local x, lorsque celui-ci est identifié, en modifiant sa stratégie de recherche. Plus précisément, le voisinage d'un optimum local $V^*(x)$est construit en se restreignant aux solutions voisines qui ne sont pas tabou, c'est-à-dire qui ne dérivent pas de l'optimum local par des mouvements qui auraient déjà permis la construction de cet optimum.

5.2.2 Liste tabou

La recherche tabou modifie localement une solution de manière itérative, tout en gardant une trace de ces modifications pendant un certain laps de temps afin d'éviter un comportement cyclique. Les modifications apportées à une solution courante deviennent donc tabou pour les solutions suivantes pendant une durée donnée, et sont enregistrées dans la liste des mouvements tabous. La liste des mouvements tabous est couramment nommée « Liste des tabous» ou « Liste tabou ». La durée pendant laquelle un mouvement reste dans l'état tabou, qui se mesure généralement en un nombre d'itérations à définir, prend le nom de « tenure ».

En général, la liste tabou contient des attributs. Un attribut qui vient d'être rendu tabou le reste pendant quelques itérations (tabu tenure), autrement dit il le sera pendant une durée qui est fixée par un ou plusieurs paramètres. Par la suite, le mouvement perd son statut de mouvement tabou, on parle alors de stratégie de diversification à court terme. Normalement, la liste

taboue doit garantir l'absence de cycles de petite taille.

5.2.3 Critère d'aspiration

Dans certains cas, les interdictions occasionnées par la liste taboue peuvent être jugées trop radicales. En effet, on risque d'éliminer (en les rendant tabous), certains mouvements particulièrement utiles, pour y remédier, il est question d'assouplir le mécanisme gérant la liste taboue.

Un mécanisme d'aspiration détermine un critère selon lequel un mouvement, bien qu'il soit tabou, peut quand même être accepté. Cependant, il va falloir faire attention au risque d'introduire à nouveau des cycles dans la recherche.

Ainsi, supposons que lors de la construction du voisinage de l'optimum local de la solution s, la solution $s\prime$ se trouve dans un état tabou. La solution $s\prime$ ne devrait théoriquement donc pas être retenue. Avec la notion de « critère d'aspiration », il est cependant possible que s' soit la meilleure solution trouvée, si par exemple $f(s\prime)$ est meilleure que toutes les autres solutions du voisinage.

5.2.4 Liste taboue : adaptation au problème à résoudre

Les principes énoncés ci-dessus doivent être adaptés au cas par cas au problème traité. Il demeure entendu que c'est au concepteur de l'algorithme de spécifier le mécanisme tabou, selon le problème traité. Aussi, pour un problème donné, plusieurs techniques différentes sont concevables. Le mécanisme

de liste taboue présenté représente réellement une seule possibilité parmi d'autres techniques qui seraient également possibles.

5.2.5 Spécification du mécanisme de liste taboue

Un mécanisme tabou est décrit par la structure de la liste taboue (la nature des éléments qui la composent) et de la manière par laquelle la liste est mise à jour quand un mouvement est effectué (quels éléments sont ajoutés) ainsi que les mouvements ou voisins interdits, selon le contenu de la liste.

Il faudra par ailleurs déterminer pour combien de temps un élément devra rester tabou.

5.2.6 Techniques additionnelles

Un algorithme tabou de base comprend une liste taboue (mémoire à court terme) ainsi qu'un critère d'aspiration. Un algorithme tabou «évolué» comprend en outre une technique de diversification et / ou une technique d'intensification. Les techniques de diversification et d'intensification font appel à des mémoires à long terme.

5.2.7 Mémoire à long terme

Dans la méthode Tabou, on peut utiliser des mémoires à long terme. Celles-ci peuvent servir à implanter des techniques d'intensification et/ou de diversification. Les mémoires à long terme permettent de stocker des solutions entières -par exemple, des solutions particulièrement performantes (élite so-

lutions), elles permettent &également de stocker la fréquence selon laquelle un attribut a appartenu à la solution courante et la fréquence selon laquelle un mouvement a été effectué, ou selon laquelle un attribut a été réintroduit dans la solution.

5.2.8 Solution d'élite

Une solution d'élite est une solution mémorisée, et qui représente une des meilleures solutions rencontrées au cours de la recherche.

5.2.9 Intensification

L'idée à la base de l'intensification est qu'on devrait explorer de façon plus approfondies les régions qui semblent les plus prometteuses.

Le principe de l'intensification consiste à retourner périodiquement visiter des zones de l'espace de recherche qui semblent particulièrement prometteuses.

De nombreuses techniques ont été proposées :

-Repartir de bonnes solutions déjà rencontrées ;

-Reconstruire une solution de départ qui tente de combiner des attributs qui ont été présents souvent dans les configurations visitées ;

-Geler certains attributs qui ont été souvent présents dans les configurations visitées ou dans les configurations d'élite relevées.

Techniques d'intensification

- Technique 1 :

Redémarrer la recherche à partir d'une solution d'élite et geler les composants qui semblent les plus intéressants.

- Technique 2 :

Redémarrer la recherche à partir d'une solution d'élite et changer le voisinage pour autoriser davantage de types de mouvements, de manière à élargir le voisinage. Si une liste de candidats est utilisée, augmenter l'échantillon.

5.2.10 Diversification

C'est un mécanisme par lequel on arrête le cours habituel de la recherche et on oblige temporairement la recherche à se diriger vers des régions inexplorées.

Diversification par relance : On construit une solution qui contient des composants rarement utilisés, et on effectue une relance à partir de cette solution.

Diversification en continu : On biaise l'évaluation des mouvements en ajoutant à l'objectif un terme relié à la fréquence des attributs :

- Les attributs les plus fréquents sont pénalisés.
- Les attributs les moins fréquents sont encouragés.

Techniques de diversification

Le principe de la diversification consiste à inciter l'algorithme à se diriger vers des régions qui n'ont pas encore été visitées.

De nombreuses techniques ont été proposées :

- Repartir d'une configuration aléatoire.

- Reconstruire une solution de départ qui tente de combiner des attributs qui ont été présents le moins souvent dans les configurations visitées.

- Modifier la fonction coût pour favorier les régions peu visitées.

Difficulté typiquement rencontrée : la recherche est «trop locale» et reste confinée dans une portion de l'espace de recherche. On risque donc d'omettre d'explorer certaines régions contenant de bonnes solutions.

5.2.11 Oscillations stratégiques

L'oscillation stratégique consiste à utiliser l'intensification et la diversification malgré qu'elles visent des objectifs opposés. L'intensification n'est pas forcément nécessaire, car on peut estimer que les régions visitées ont été suffisamment exploré. La diversification est généralement plus importante, car il est fréquent que la recherche reste confinée dans certaines régions.

5.2.12 Critère d'arrêt

Un critère d'arrêt est une condition qui –lorsqu'elle est vérifiée ou atteinte- permet de mettre fin à la recherche.

Les conditions suivantes peuvent bien constituer des critères d'arrêt :

- Si une solution prouvée optimale a été trouvée.

- Si une limite a été atteinte.

- Le nombre d'itérations.

- Le temps de calcul.

- Si la recherche semble stagner : nombre d'itérations sans amélioration de la meilleure configuration trouvée.

5.2.13 Listes de candidats

Dans la recherche avec tabou, nous sommes sensé de chercher le meilleur mouvement non tabou. Cependant, cette manière de procéder peut se révéler trop coûteuse.

Pour y remédier, nous pouvons se limiter à engendrer seulement un sous-ensemble des voisins (ou mouvements). On parle de liste de candidats.

Les candidats de la liste peuvent être engendrés de manière aléatoire (échantillon aléatoire). Une autre approche consiste à sélectionner les voisins qui semblent les plus prometteurs – selon un critère quelconque.

5.2.14 Schéma de l'algorithme tabou de base (pour un problème de minimisation)

Algorithme 28 *Tabou de Base*

Input: *Une configuration initiale* S_0

Output: *Meilleur solution rencontrée* S^*

$S := S_0$

$S^* := S$

liste tabouеT := $\{\varnothing\}$

Tant que <*Critère d'arrêt non atteint*> ***faire***

liste des Candidats := $\{\varnothing\}$

Pour $S_{candidat} \in S_{meilleur\ du\ voisinage}$ ***faire***

Si $S_{candidat} \notin \{$*Solutions considérées comme tabou*$\}$ ***alors***

liste des Candidats := $\{S_{candidat}\}$

Fin Si

Fin Pour

$S_{candidat}$:= *Meilleur solution de liste des Candidats*

Si $f(S_{candidat}) \leq f(S^*)$ ***alors***

$S^* := S_{candidat}$

Si $S_{candidat} \notin \{$*Solutions considérées comme tabou*$\}$ ***alors***

liste des Candidats := $\{S_{candidat}\}$

Fin Si

Mettre T à jour

Fin Tant que

Retourner S^*

5.2.15 Organigramme Général de la Recherche Tabou

Ci-dessous une représentation de l'organigramme général de la Recherche Tabou :

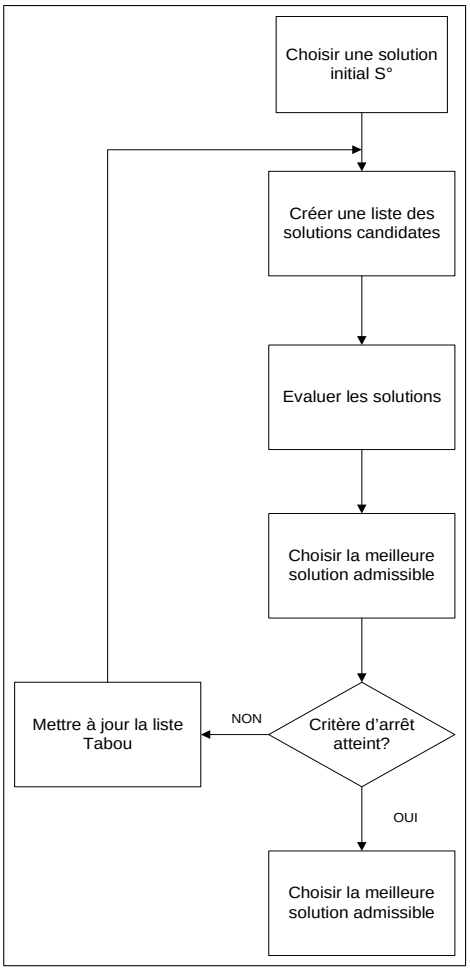


Figure 1: Organigramme général de la Recherche Tabou

5.3 Adaptation de la méthode Tabou au Problèmes des flots maximaux

5.3.1 Espace de recherche

La recherche de nouvelles solutions améliorant la valeur de la fonction objectif se fera sur l'ensemble des solutions efficaces du problème (MC), à savoir $X_{E.}$ Pour rappel, le problème (MC) s'écrit sous la forme :

$$(MC) \quad \left| \begin{array}{ll} \text{vecteur à maximiser} & Cx \\ \text{sujet à} & x \in X \end{array} \right.$$

Ainsi, une configuration S est l'ensemble des solutions efficaces issues de la résolution du problème (MC) à l'aide de La Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes « SEEVD »

5.3.2 Fonction d'évaluation

Il s'agit du problème du minimax flow ; le but étant de minimiser une fonction objectif sur l'ensemble des solutions efficaces obtenu en résolvant le problème (MC), en maximisant au préalable la valeur du flot passant par chaque arc. Comme fonction d'évaluation, nous prendrons comme fonction à minimiser :

$$f(x) = \sum_{h=1}^{|E|} x_h.$$

5.3.3 Voisinage

Un mouvement consiste à passer d'une solution x à une autre solution voisine x' améliorant d'avantage la valeur de la fonction objectif. Le mouvement sera décrit alors par un couplet (x, x') où : $f(x') \leq f(x)$.

Le choix d'une solution voisine se fera sur une base d'une sélection aléatoire.

5.3.4 Mise à jour de la liste

La liste tabou sera mise à jour lorsque un mouvement (x, x') est effectué, en ajoutant à la liste l'élément (x, x').

5.3.5 Mouvements interdits

Si le couple (x, x') appartient à la liste, cela interdit d'ores le mouvement $< x, x' >$. Le mécanisme décrit revient, quand le mouvement $< x, x' >$ est effectué, à interdire à x de retrouver son ancienne valeur $f(x)$. Autrement dit, quand on effectue le mouvement $< x, x' >$, on interdit le mouvement inverse $< x, f(x) >$.

5.4 Algorithme de Résolution des Problèmes des flots maximaux

Algorithme 29 *Résolution des Problèmes des flots maximaux*

Input: *Un réseau* $R = (V, s, t, \partial^+ h, \partial^- h, c_h)$

Output: *Un flot maximal minmisé* X^*

1−*Appliquer la Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes* ***« SEEVD »*** *sur le problème associé (PLE)*

2− *Choisir aléatoirement une solution de départ* $X^0 \in X_E$

liste tabou$eT := \{\varnothing\}$

Tant que $<$*Critère d'arrêt non atteint*$>$ ***faire***

liste des Candidats $:= \{X^0\}$

Pour $X_{candidat} \in X_{meilleur\ du\ voisinage}$ ***faire***

Si $X_{candidat} \notin \{$*Solutions considérées comme tabou*$\}$ ***alors***

liste des Candidats $:= \{X_{candidat}\}$

Fin Si

Fin Pour

$X_{candidat} :=$ *Meilleur solution de liste des Candidats*

Si $f(X_{candidat}) \leq f(X^*)$ ***alors***

$X^* := X_{candidat}$

Si $X_{candidat} \notin \{$*Solutions considérées comme tabou*$\}$ ***alors***

liste des Candidats := $\{X_{candidat}\}$

Fin Si

Mettre T à jour

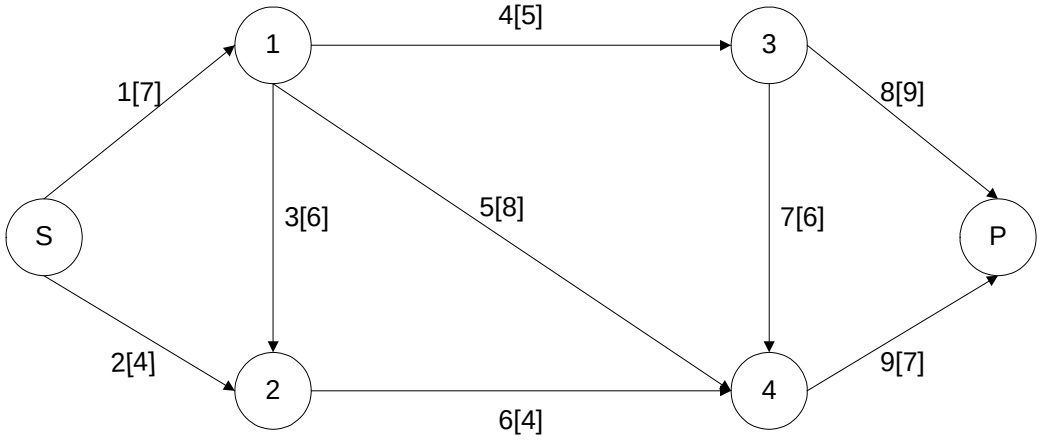
Fin Tant que

Retourner X^*

6 Exemple d'application

6.1 Formulation

Nous allons dans ce qui suit procéder à l'application des méthodes citées auparavant sur un exemple concret. Nous considérons un réseau $R = (X, s, p, U, c_h)$ avec $|X| = 4, |U| = 9$; soit alors le graphe suivant :



où s est le sommet source et p est le sommet puits, les capacités sont défénies de la manière suivante :

$$c_h = \text{capacité de l'arc } h.$$

Soit alors les capacités suivantes :

$c_1 = 7$; $c_2 = 4$; $c_3 = 6$; $c_4 = 5$; $c_5 = 8$; $c_6 = 4$; $c_7 = 6$; $c_8 = 9$; $c_9 = 7$;

La matrice d'incidence sommets arcs se présentera comme suit :

$$A = \begin{bmatrix} -1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 1 \end{bmatrix}$$

Ainsi,le programme linéaire associé au problème du flot maximal mono-objectif se décrira comme suit :

6.1.1 Contraintes de conservation

$$-x_1 + x_3 + x_4 + x_5 = 0$$
$$-x_2 - x_3 - x_6 = 0$$
$$-x_4 + x_7 + x_8 = 0$$
$$-x_5 - x_6 - x_7 + x_9 = 0$$

6.1.2 Contraintes de capacité

$$x_1 <= 7$$
$$x_2 <= 4$$
$$x_3 <= 6$$
$$x_4 <= 5$$
$$x_5 <= 8$$
$$x_6 <= 4$$
$$x_7 <= 6$$

$$x_8 <= 9$$

$$x_9 <= 7$$

$$x_1; x_{2;} x_3; x_4; x_5; x_6; x_7; x_8; x_9 \in \mathbb{N}$$

6.1.3 Fonction objectif

L'objectif étant de maximiser la valeur du flot. La fonction objectif sera alors décrite comme suit :

$$Z_{max} = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9.$$

6.1.4 Programme linéaire associé au problème du flot maximum

Voici donc le programme linéaire associé au problème du flot maximum :

$$(PL) = \begin{cases} -x_1 + x_3 + x_4 + x_5 = 0 \\ -x_2 - x_3 - x_6 = 0 \\ -x_4 + x_7 + x_8 = 0 \\ -x_5 - x_6 - x_7 + x_9 = 0 \\ x_1 <= 7 \\ x_2 <= 4 \\ x_3 <= 6 \\ x_4 <= 5 \\ x_5 <= 8 \\ x_6 <= 4 \\ x_7 <= 6 \\ x_8 <= 9 \\ x_9 <= 7 \\ x_1; x_2; x_3; x_4; x_5; x_6; x_7; x_8; x_9 \in \mathbb{N} \\ Z_{max} = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 \end{cases}$$

6.1.5 Programme linéaire associé au problème du flot maximal

Le programme linéaire associé au problème du flot maximal comportera les contraintes du programme linéaire précédent ; tout en optimisant le vecteur critère : $MaxZ_k(x) = x_k \quad k = 1..|E|$ soit :

$$
PLE = \begin{cases}
-x_1 + x_3 + x_4 + x_5 = 0 \\
-x_2 - x_3 - x_6 = 0 \\
-x_4 + x_7 + x_8 = 0 \\
-x_5 - x_6 - x_7 + x_9 = 0 \\
x_1 <= 7 \\
x_2 <= 4 \\
x_3 <= 6 \\
x_4 <= 5 \\
x_5 <= 8 \\
x_6 <= 4 \\
x_7 <= 6 \\
x_8 <= 9 \\
x_9 <= 7 \\
x_1; x_{2;} x_3; x_4; x_5; x_6; x_7; x_8; x_9 \in \mathbb{N} \\
MaxZ_1 = x_1 \\
MaxZ_2 = x_2 \\
MaxZ_3 = x_3 \\
MaxZ_4 = x_4 \\
MaxZ_5 = x_5 \\
MaxZ_6 = x_6 \\
MaxZ_7 = x_7 \\
MaxZ_8 = x_8 \\
MaxZ_9 = x_9
\end{cases}
$$

6.2 Résolution

Le programme linéaire précédent se présente sous la forme du tableau décrit dans la page suivante lors de l'implémentation dans le programme de résolution.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	$Type$	RHS
contrainte (1)	-1	0	1	1	1	0	0	0	0	$=$	0
contrainte (2)	0	-1	-1	0	0	1	0	0	0	$=$	0
contrainte (3)	0	0	0	-1	0	0	1	1	0	$=$	0
contrainte (4)	0	0	0	0	-1	-1	-1	0	1	$=$	0
contrainte (5)	1	0	0	0	0	0	0	0	0	$\leq$	7
contrainte (6)	0	1	0	0	0	0	0	0	0	$\leq$	4
contrainte (7)	0	0	1	0	0	0	0	0	0	$\leq$	6
contrainte (8)	0	0	0	1	0	0	0	0	0	$\leq$	5
contrainte (9)	0	0	0	0	1	0	0	0	0	$\leq$	8
contrainte (10)	0	0	0	0	0	1	0	0	0	$\leq$	4
contrainte (11)	0	0	0	0	0	0	1	0	0	$\leq$	6
contrainte (12)	0	0	0	0	0	0	0	1	0	$\leq$	9
contrainte (13)	0	0	0	0	0	0	0	0	1	$\leq$	7
Z_1	1	0	0	0	0	0	0	0	0	max	
Z_2	0	1	0	0	0	0	0	0	0	max	
Z_3	0	0	1	0	0	0	0	0	0	max	
Z_4	0	0	0	1	0	0	0	0	0	max	
Z_5	0	0	0	0	1	0	0	0	0	max	
Z_6	0	0	0	0	0	1	0	0	0	max	
Z_7	0	0	0	0	0	0	1	0	0	max	
Z_8	0	0	0	0	0	0	0	1	0	max	
Z_9	0	0	0	0	0	0	0	0	1	max	

L'application de la Méthode de détermination des Solutions Efficaces dans l'Espace des Variables Discrètes « SEEVD » sur le problème (PLE) engendre l'ensemble des slolutions efficaces suivant :

$X_E = \{(7,4,0,5,2,4,1,4,7),(7,2,2,5,0,4,3,2,7),(7,3,1,5,1,4,2,3,7),$
$(7,4,0,4,3,4,0,4,7),(7,4,0,5,2,4,0,5,6),(5,4,0,5,0,4,3,2,7),$
$(6,4,0,5,1,4,2,3,7),(7,0,0,0,7,0,0,0,7),(7,1,0,1,6,1,0,1,7),$
$(7,2,0,2,5,2,0,2,7),(7,3,0,3,4,3,0,3,7),(7,3,1,5,1,4,1,4,6),$
$(7,3,1,5,1,4,0,5,5),(7,3,1,4,2,4,1,3,7),(7,3,0,5,2,3,2,3,7)\}$.

L'ensemble des solutions efficaces du problème (PLE) correspond à l'ensemble des critères non dominés, autrement dit $X_E = Z_{ND}$:

$Z_{ND} = \{(7,4,0,5,2,4,1,4,7),(7,2,2,5,0,4,3,2,7),(7,3,1,5,1,4,2,3,7),$
$(7,4,0,4,3,4,0,4,7),(7,4,0,5,2,4,0,5,6),(5,4,0,5,0,4,3,2,7),$
$(6,4,0,5,1,4,2,3,7),(7,0,0,0,7,0,0,0,7),(7,1,0,1,6,1,0,1,7),$
$(7,2,0,2,5,2,0,2,7),(7,3,0,3,4,3,0,3;7),(7,3,1,5,1,4,1,4,6),$
$(7,3,1,5,1,4,0,5,5),(7,3,1,4,2,4,1,3,7),(7,3,0,5,2,3,2,3,7)\}$.

6.2.1 Ajustement des paramètres de la Recherche Tabou

- **Taille de la liste Tabou :** la taille de la liste Tabou a été ajustée à 4 éléments.
- **Critère d'aspiration :** Nous définissons comme critère d'aspiration le faite qu'un mouvement tabou quelconque, ne peut le demeurer plus de 4 itérations.

- **Choix de la liste des candidats** : Le choix se fera d'une manière aléatoire tout en privilégiant les solutions non encore testées.

- **Taille de la liste des candidats** : La taille de la liste des candidats est arrêtée à 2.

- **Solution sélectionnée** : Parmi la liste des solutions candidates, nous choisissons celle qui améliore (minimise d'avantage) la valeur de la fonction d'évaluation.

- **Critère d'arrêt** : Nous choisissons d'arrêter l'algorithme après un certain nombre d'itérations. Pour cet exemple nous choisissons d'arrêter la recherche après 10 itérations.

  Par souci de clarté, notons les solutions efficaces déjà trouvées auparavant comme suit :

$X_1 = (7, 4, 0, 5, 2, 4, 1, 4, 7)$

$X_2 = (7, 2, 2, 5, 0, 4, 3, 2, 7)$

$X_3 = (7, 3, 1, 5, 1, 4, 2, 3, 7)$

$X_4 = (7, 4, 0, 4, 3, 4, 0, 4, 7)$

$X_5 = (7, 4, 0, 5, 2, 4, 0, 5, 6)$

$X_6 = (5, 4, 0, 5, 0, 4, 3, 2, 7)$

$X_7 = (6, 4, 0, 5, 1, 4, 2, 3, 7)$

$X_8 = (7, 0, 0, 0, 7, 0, 0, 0, 7)$

$X_9 = (7, 1, 0, 1, 6, 1, 0, 1, 7)$

$X_{10} = (7, 2, 0, 2, 5, 2, 0, 2, 7)$

$X_{11} = (7, 3, 0, 3, 4, 3, 0, 3, 7)$

$X_{12} = (7, 3, 1, 5, 1, 4, 1, 4, 6)$

$X_{13} = (7, 3, 1, 5, 1, 4, 0, 5, 5)$

$X_{14} = (7, 3, 1, 4, 2, 4, 1, 3, 7)$

$X_{15} = (7, 3, 0, 5, 2, 3, 2, 3, 7)$

6.2.2 Déroulement de l'algorithme

Ci-après les étapes du déroulement de l'algorithme de recherche :

Itération	*Candidats*	*Sélection*	*Mouvement*	*Valeur*	*Liste tabou*	*Z*
0	X_1	X_1	$-$	$-$	$\varnothing$	34

Première itération : le choix d'une solution de départ se fait d'une manière aléatoire ; l'algorithme commence par la solution initiale X_1 , l'ensemble des solutions candidates est alors égale à la solution choisie , la liste tabou est vide, la valeur de la fonction d'évaluation correspondante à X_1 est égale à 34.

Itération	*Candidats*	*Sélection*	*Mouvement*	*Valeur*	*Liste tabou*	*Z*
1	X_4, X_5	X_4	(X_1, X_4)	-1	$\{(X_1, X_4)\}$	33

La liste des candidats est construite aléatoirement en privilégiant les solutions non encore testées, en l'occurrence dans ce cas $\{X_4, X_5\}$, la solution X_4 est choisie étant donné qu'elle représente celle qui minimise d'avantage la valeur de la fonction d'évaluation.La mouvement effectué (X_1, X_4) est alors ajouté à la liste tabou, ayant une valeur de -1, la valeur de la fonction d'évaluation devient désormais 33.

Itération	*Candidats*	*Sélection*	*Mouvement*	*Valeur*	*Liste tabou*	*Z*
2	X_3, X_5	X_3	(X_4, X_3)	0	$\{(X_1, X_4); (X_4, X_3)\}$	33

La solution X_3 étant choisie, le mouvement (X_4, X_3) est ajouté à la liste tabou qui mémorise déjà le mouvement précédent (X_1, X_4), ainsi ces deux mouvements sont d'ores interdits.

Itération	*Candidats*	*Sélection*	*Mouvement*	*Valeur*	*Liste tabou*	*Z*
3	X_1, X_5	X_5	(X_3, X_5)	0	$\{(X_1, X_4); (X_4, X_3); (X_3, X_5)\}$	33
4	X_4, X_7	X_7	(X_5, X_7)	-1	$\{(X_1, X_4); (X_4, X_3); (X_3, X_5); (X_5, X_7)\}$	32
5	X_{12}, X_{15}	X_{12}	(X_7, X_{12})	0	$\{(X_4, X_3); (X_3, X_5); (X_5, X_7); (X_7, X_{12})\}$	32

Cinquième itération : le critère d'aspiration est appliqué, le premier mouvement (X_1, X_4) ayant été défini comme tabou ne le devient plus et sort de la liste tabou, désormais, si l'algorithme se retrouve à la solution X_1, le mouvement (X_1, X_4) pourra être effectué normalement.

Itération	*Candidats*	*Sélection*	*Mouvement*	*Valeur*	*Liste tabou*	*Z*
6	X_6, X_{13}	X_6	(X_{12}, X_6)	-2	$\{(X_3, X_5); (X_5, X_7); (X_7, X_{12}); (X_{12}, X_6)\}$	30
7	X_2, X_{11}	X_{11}	(X_6, X_{11})	0	$\{(X_5, X_7); (X_7, X_{12}); (X_{12}, X_6); (X_6, X_{11})\}$	30
7	X_{14}, X_{10}	X_{10}	(X_{11}, X_{10})	-3	$\{(X_7, X_{12}); (X_{12}, X_6); (X_6, X_{11}); (X_{11}, X_{10})\}$	27
9	X_5, X_8	X_8	(X_3, X_8)	-6	$\{(X_{12}, X_6); (X_6, X_{11}); (X_{11}, X_{10})\}; (X_3, X_8)\}$	21
10	X_{12}, X_9	X_9	(X_8, X_9)	$+3$	$\{(X_6, X_{11}); (X_{11}, X_{10})\}; (X_3, X_8); (X_8, X_9)\}$	24

Dernière itération : dernière solution choisie X_9, celle-ci est sélectionnée en dépit de sa détérioration de la valeur de la fonction d'évaluation, c'est un des avantages de l'algorithme de la recherche tabou, permettant beaucoup plus de souplesse, et surtout de ne pas être coincé dans un minimum local.

Finalement, la solution retenue sera celle trouvée à l'étape 9 , à savoir $X^* = X_8$, la valeur de la fonction d'valuation correspondante est égale à $Z^* = 21$

Ci-apres le graphe representatif des valeurs de la fonction d'évaluation en fonction des solutions efficaces :

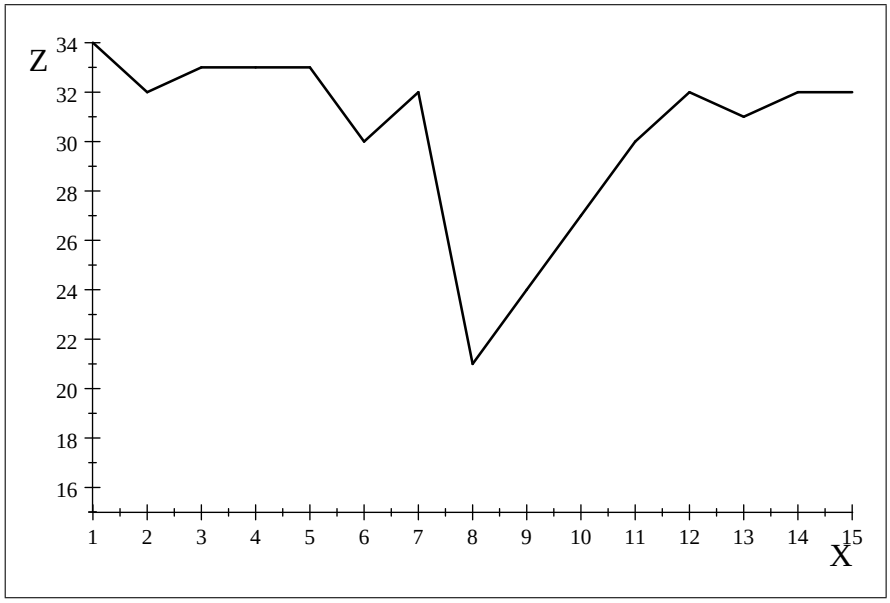


Figure 3 : Valeurs de Z

et ci-dessious le graphe representant l'évolution des valeurs de Z pendant le dérouelemnt de l'algorithme:

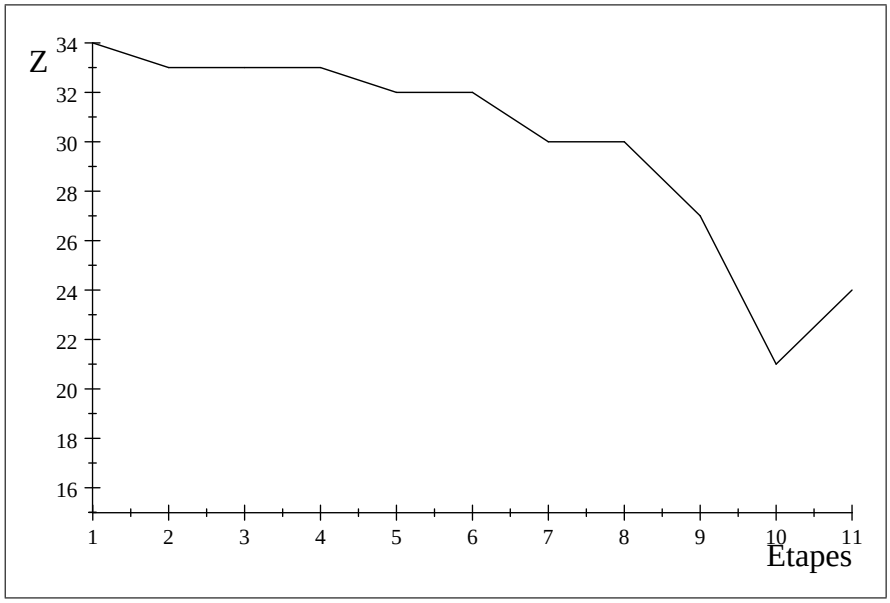


Figure 4 : Evolutions des valeurs de Z durant l'éxecution de l'algorithme.

7 Conclusion :

L'optimisation multi-objectifs n'a cessé d'être un domaine attirant en raison des innombrables issues et possibilités qu'elle offre pour tout problème du genre. Il en est de même des métaheuristiques permettant de faire face aux problèmes liés à la complexité algorithmique de certaines méthodes exactes.

La mise en application des méthodes de résolution est souvent sujette à des difficultés d'ordre technique, conséquence des tailles importantes des problèmes traités, de la complexité des algorithmes de résolutions, ainsi que les limites techniques que présentent les machines de calcule. C'est ainsi que les recherches prirent la voie des méthodes hybrides, impliquant souvent les métaheuristiques.

Il a été exposé dans ce travail un type particulier de problèmes d'optimisation multi-objectifs, dont la méthode de résolution s'est construite moyennant une approche combinant une méthode exacte et une métaheuristique. En dépit de l'extrême sensibilité des problèmes des flots maximaux multi-objectifs vis-à-vis l'aspect taille. Il serait captivant de tirer profit des avantages qu'offre cette dernière pour opter à son application sur d'autres substances dont la taille sera considérée comme très grande.

Bibliographie

[1] A. Abraham, J. Jain, and R Goldberg. *Evolutionary Multiobjective Optimization : Theoretical Advances and Applications*. Springer Science+Business Media, LLC, 2005.

[2] B. Bachelet. *Recherche Opérationnelle*. 2003.

[3] A. Berro. *Optimisation Multiobjectif et Stratégies d'évolution en Environnement Dynamique*, pages 27–72. Université des Sciences Sociales Toulouse I, 2001.

[4] J Branke, K. Miettinen, K. Deb, and R. Slowinski. *Multiobjective Optimization Interactive and Evolutionary Approaches.*, pages 1–52. Springer Science+Business Media, LLC, 2008.

[5] D. Chaabane. *Contribution à l'Optimisation Multicritère en Variables Discrètes*, pages 954–965. Faculté Polytechnique de Mons, 2007.

[6] Y. Collette and P. Siarry. *Optimisation Multiobjectif*, pages 1–129. Éitions EYROLLES 61, Bld Saint-Germain 75240 Paris Cedex 05, 2002.

[7] D. De Wolf. *Recherche Opérationnelle*, pages 69–7. Université du Littoral, 2003.

[8] D. Dodge. *Optimisation appliquée*, pages 89–90. Springer Science+Business Media, LLC, 2005.

[9] K.F. Doerner and al. *Metaheuristics: Progress in Complex Systems Optimization*, pages 43–60;325–342. Springer Science+Business Media, LLC, 2007.

[10] Y. Donoso and R. Fabregat. *Multi-Objective Optimization in Computers Networks Using Metaheuristics*, pages 32–34;44–45. Auerbach Publications, 2007.

[11] M. Ehergott. *Multicriteria Optimization.* Springer Science+Business Media, LLC, 2005.

[12] R. Faure, B. Lemaire, and C. Picouleau. *Précis de Recherche Opérationnelle*, pages 80–87. Dunod, 2009.

[13] D.Y Gao and H.D. Sheral. *Advances In Applied Mathematics And Global Optimization*, pages 137–193. Springer Science+Business Media, LLC, 2009.

[14] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, 1997.

[15] B. Golden, S. Raghavan, and E. Wasil. *The Next Wave In Computing Optimization and Decision Technologie.* Springer Science+Business Media, LLC, 2005.

[16] F.S Hillier and G.J. Lieberman. *Introduction To Operations Research*, pages 420–434. McGraw-Hill Series in Industrial Engineering and Management Science, 2001.

[17] J.H. Hooker. *Integrated Methods for Optimization*, pages 88–94. Springer Science+Business Media, LLC, 2007.

[18] T. Ibaraki, K. Nonobe, and M. Yagiura. *Metaheuristics: Progress as Real Problem Solvers*, pages 65–84. Springer Science+Business Media, LLC, 2005.

[19] L. Kandille. *Principles of Mathematics in Operations Research*, pages 103–111. Springer Science+Business Media, LLC, 2007.

[20] J. Knowles, D. Corne, and K. Deb. *Multiobjective Problem Solving from Nature : From Concepts to Applications.* Springer Science+Business Media, LLC, 2008.

[21] B. Korte and J. Vygen. *Optimisation Combinatoire : Théorie et Algorithmes*, pages 171–193. Springer Science+Business Media, LLC, 2010.

[22] M. Moulay. *Support du cours : La Programmation Mathématique Multicritère, La Programmation Linéaire Multicritère En Variables Entières (PLME)*, pages 7–21;61–91. 2006.

[23] J. Philip. *Algorithms for the vector maximization problem*, pages 207–229. Mathematical Programming, 1972.

[24] M. Raspaud. *Recherche Opérationnelle*, pages 33–43. 2006.

[25] P. Siarry and Z. Michalewicz. *Advances in Metaheuristics for Hard Optimization*, pages 69–83; 329–350. Springer Science+Business Media, LLC, 2009.

[26] G. Strang. *Advances In Applied Mathematics And Global Optimization : Maximum Flows and Minimum Cuts in the Plane*, pages 1–9. Springer Science+Business Media, LLC, 2009.

[27] J.E. Tahir, M.A. andSmith. *Feature Selection For Heterogeneous EnsemblesOf Nearest-Neighbour Classifers Using Hybrid Tabu Search.* Springer Science+Business Media, LLC, 2008.

[28] Y. Yamamoto. *Optimization Over The Efficient Set : Overview.* Kluwer Academic Publishers, 2001.

[29] Y. Yamamoto. *Minimum Maximal Flow Problem An Optimization Over The Efficient Set.* 2007.

[30] Y. Yamamoto and D. Zenke. *D.C. Optimization Methods for Minimum Maximal Flow Problem.* University Of TsukubTsukuba, Ibaraki 305-8573 JAPAN, 2004.