UNIVERSITÉ DE NANCY I

THÈSE

présentée à

L'UNIVERSITÉ DE NANCY I

pour obtenir le grade de

DOCTEUR ÈS-SCIENCES

par

Mohamed Yehia YOUSSEF

SUJET :

METHODES ET PROGRAMMES D'OPTIMISATION DANS LES PROBLEMES NON LINEAIRES DE FLOT ET DE TRANSPORT

Soutenue publiquement le 2 Octobre 1978 devant la Commission d'Examen

Membres du Jury :

Président : M. J. LEGRAS

Examinateurs : MM. P. BOYER
M. CASTAGNÉ
Cl. GILORMINI
J.P. HATON

UNIVERSITÉ DE NANCY I

THÈSE

Mohamed Yehia YOUSSEF

SUJET :

METHODES ET PROGRAMMES D'OPTIMISATION DANS LES PROBLEMES NON LINEAIRES DE FLOT ET DE TRANSPORT

A la Mémoire de Mes Parents.

L'étude qui fait l'objet de la présente thèse a été effectuée sous la direction de Monsieur le Professeur J. LEGRAS, de l'Université de NANCY I à qui j'exprime toute ma reconnaissance et mon respectueux attachement, pour la bienveillance et l'aide incessante qu'il m'a toujours prodiguées.

C'est grâce à ces conseils, ses encouragements, sa bonté que ce travail a pu être réalisé.

Je remercie très vivement Monsieur le Professeur C. GILORMINI et Monsieur M. CASTAGNÉ, Maître de Conférences pour avoir accepté d'être mes rapporteurs, et de me faire l'honneur d'être membres du Jury.

Que Monsieur J. P. HATON et Monsieur P. BOYER soient assurés de ma reconnaissance pour avoir bien voulu examiner mon travail et me faire l'honneur de participer au Jury de cette thèse.

Mes remerciements vont aussi à Monsieur POTDEVIN et Monsieur SANCHEZ du Laboratoire d'Informatique de l'Université de NANCY I, pour leurs précieux conseils et pour le temps qu'ils ont bien voulu me consacrer.

Je remercie également Madame D. MARCHAND pour le soin et la gentillesse avec lesquels elle s'est chargée de la réalisation matérielle de cette thèse.

Enfin, j'exprime ma très sincère amitié aux chercheurs Français et Etrangers, qui travaillent sous la direction de Monsieur le Professeur LEGRAS.

ERRATA

Page 6

Entre la cinquième et la 6ème ligne en bas de la page, on ajoute :

$$C_\ell (X^r + h \,.\, D) = 0 \qquad \forall\, \ell \in L \;;$$

$$C_j (X^r + h \,.\, D) \geq 0 \qquad \forall\, j \in J.$$

Page 12

Troisième ligne en bas lire :

le calcul à partir de la première phase de la première étape.

au lieu de

le calcul à partir de la première phase de la deuxième étape.

Page 98

Cinquième ligne

$$\sum_{i=1}^{3} q_i > \sum_{j=1}^{10} Q_j$$

au lieu de

$$\sum_{i=1}^{3} P_i > \sum_{j=1}^{10} R_j.$$

TABLE DES MATIERES

NOMENCLATURE DES SYMBOLES

$X \equiv \{x_1, x_2, \ldots, x_n\}$: Un domaine admissible d'un espace E^n. Les inconnues du problème à optimiser.

$f(X)$: Fonction objectif non linéaire.

L : L'ensemble des indices des contraintes bilatérales.

J : L'ensemble des indices des contraintes unilatérales saturées.

K : L'ensemble des indices des contraintes unilatérales vérifiées non saturées.

$C_\ell(X)$: La contrainte bilatérale numéro ℓ ; $\ell \in L$.

$C_j(X)$: La contrainte unilatérale vérifiée numéro j.

λ_ℓ : Coefficients de Lagrange. A l'optimum, ils portent le nom de multiplicateurs de Lagrange.

λ_j : Multiplicateurs de Kuhn et Tucker.

D : Une direction de montée admissible.

A : Une matrice dont chacun de ses coefficents soit 0, 1, ou -1.

$\mathscr{C}'$: La matrice des dérivées des contraintes bilatérales et unilatérales par rapport aux variables différentes.

QE : Une quantité de flot entrante dans un graphe.

G : Un graphe orienté et évalué.

X_B^r : Les variables de base à l'itération r.

X_H^r : Les variables hors base à l'itération r.

$p(u)$: La capacité de l'arc u dans le graphe G.

INTRODUCTION

L'optimisation du coût dans un problème de flot ou de transport est classique en programmation linéaire, mais a été peu étudiée lorsque le coût est fonction non linéaire des variables.

R. BELLMAN [1] a donné une solution à ce problème non linéaire par une méthode d'approximations successives, méthode dont on n'est pas assuré de la convergence et qui devient extrêmement lourde lorsque le nombre de sources augmente.

Nous nous proposons d'étendre aux problèmes de flot et de transport non linéaires la méthode classique du gradient projeté en apportant aux programmes de notre bibliothèque d'optimisation les modifications et simplifications dûes à la spécificité de cette classe de problème où la fonction objectif est non linéaire, mais où les contraintes sont très particulières : elles sont évidemment linéaires et de plus la matrice des dérivées des contraintes est unimodulaire.

Ces modifications seront exposées dans la première partie de ce travail. La seconde partie consiste en :

- une application sur un problème de flot ;

- un exemple d'un problème de transport déjà traité par R. BELLMAN en utilisant la méthode d'approximations successives ; ceci nous permet de comparer les deux solutions, ainsi que les valeurs de la fonction objectif optimale dans les deux cas.

- Un exemple d'un problème de transport général, c'est-à-dire dans le cas où l'offre dépasse la demande.

PARTIE I

I - 1 RAPPEL DES PROPRIETES D'UN OPTIMUM D'UNE FONCTION NON LINEAIRE DE N-VARIABLES NON INDEPENDANTES

Dans la suite on cherche le maximum d'une fonction non linéaire $f(x_1, x_2, \ldots, x_n)$ dans un domaine admissible d'un espace E^n. Deux cas principaux peuvent se présenter :

i) le domaine admissible représenté par l'ensemble $X = \{x_1, x_2, \ldots x_n\}$ est astreint à vérifier l'ensemble des contraintes bilatérales :

$$C_\ell (X) = 0 \qquad \ell = 1, 2, \ldots, p \text{ et } p < n \qquad (1)$$

Une condition nécessaire pour que f admette un maximum à $X^* = \{x_1^*, x_2^*, \ldots, x_n^*\}$ vérifiant les contraintes bilatérales (1) est qu'il existe des nombres λ_ℓ appelés "coefficients de Lagrange" ; et tels que soient vérifiées les n-équations :

$$\frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_i} = \sum_{\ell=1}^{p} \lambda_\ell \frac{\partial C_\ell (x_1, x_2, \ldots, x_n)}{\partial x_i} \quad i = 1, 2, \ldots, n$$

A l'optimum $\lambda_1, \lambda_2, \ldots, \lambda_p$ portent le nom de multiplicateurs de Lagrange. Ces derniers et les coordonnées de l'optimum doivent vérifier :

- les n-équations :

$$\frac{\partial f(x_1^*, x_2^*, \ldots, x_n^*)}{\partial x_i} = \sum_{\ell=1}^{p} \lambda_\ell \frac{\partial C_\ell (x_1^*, x_2^*, \ldots, x_n^*)}{\partial x_i} \quad i=1, 2, \ldots, n$$

- les p-contraintes :

$$C_\ell (x_1^*, x_2^*, \ldots, x_n^*) = 0 \qquad \ell = 1, 2, \ldots, p.$$

ii) Le domaine admissible est astreint à vérifier l'ensemble des contraintes unilatérales :

$$C_j(X) \geq 0 \qquad j = 1, 2, \ldots, q \qquad (2)$$

Dans ce cas, on peut appliquer le théorème de Kuhn et Tucker qui s'énonce comme le suivant : [11] p. 8

"Si l'ensemble $X^* = \{x_1^*, x_2^*, \ldots, x_n^*\}$ est optimal et vérifie les contraintes (2), il existe, associés à ces derniers, q-nombres $\{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ (multiplicateurs de Kuhn-Tucker) vérifiant les conditions :

$$\left.\begin{array}{l} \lambda_j \leq 0 \\ \lambda_j \cdot C_j(x_1^*, x_2^*, \ldots, x_n^*) = 0 \end{array}\right\} \quad j = 1, 2, \ldots, q$$

et tels que :

$$\frac{\partial f(x_1^*, x_2^*, \ldots, x_n^*)}{\partial x_i} = \sum_{j=1}^{q} \lambda_j \frac{\partial C_j(x_1^*, x_2^*, \ldots, x_n^*)}{\partial x_i} \qquad i = 1, 2, \ldots, n$$

Notons qu'il est facile d'étendre ces prorpiétés au cas où nous avons simultanément des contraintes bilatérales et unilatérales, comme nous le verrons aux paragraphes suivants.

I - 2 CARACTERISATION DE L'OPTIMUM :

I.2.1 Théorème de Farkas-Minskowski [18]

Dans un espace E^n, on considère :

1) une forme linéaire et donnée W (X)
2) un groupe g_1 de p-formes linéaires U_ℓ (X)
3) un groupe g_2 de q-formes linéaires V_j (X)
4) l'ensemble de (p+q) formes indépendantes avec $p + q \leq n$.

Alors les conditions nécessaires et suffisantes pour que la forme W (X) soit négative ou nulle pour tout tableau de X vérifiant :

$$U_\ell(X) = 0 \qquad \forall \ell \in L$$

$$V_j(X) \geq 0 \qquad \forall j \in J$$

sont qu'il existe des nombres λ_ℓ et μ_j tels que :

$$W(X) = \sum_{\ell \in L} \lambda_\ell U_\ell(X) + \sum_{j \in J} \lambda_j V_j(X)$$

avec

$$\lambda_j \leq 0 \qquad \forall j.$$

I.2.2 L'optimum local d'une fonction sous contraintes mixtes :

Soit X un tableau quelconque admissible voisin des coordonnées du maximum X^*, qui vérifie :

$$\left.\begin{array}{l} f(X^*) \geq f(X) \\ C_\ell(X) = 0 \\ C_j(X) \geq 0 \\ C_k(X) \geq 0 \end{array}\right\} \qquad (3)$$

De plus X^* vérifie :

$$C_\ell(X^*) = 0 \qquad \forall \ell \in L$$

$$C_j(X^*) = 0 \qquad \forall j \in J$$

$$C_k(X^*) > 0 \qquad \forall k \in K$$

où L est l'ensemble des contraintes bilatérales,
J est l'ensemble des contraintes vérifiées saturées,
et K est l'ensemble des contraintes vérifiées non saturées.

En supposant que X est suffisamment voisin de X^*, on peut approcher les valeurs de :

1 - f (X) par :

$$f(X) = f(X^*) + f'(X^*)(X - X^*)$$

où

$$f'(X^*)(X - X^*) = \sum_{i=1}^{n} h_i f'_{x_i}(X^*).$$

2 - C_i (X) par :

$$C_i(X) = C_i(X^*) + C'_i(X^*)(X - X^*) \qquad \forall i \in \{L \cup J \cup K\}$$

où

$$C'_i(X^*)(X - X^*) = \sum_{s=1}^{n} \frac{\partial C_i}{\partial x_s}(X^*)(X_s - X^*_s)$$

et les relations (3) deviennent :

$$f(X^*) \geq f(X) \sim f(X^*) + f'(X^*)(X - X^*)$$

soit

$$f'(X^*)(X - X^*) \leq 0$$

$$C'_\ell(X^*)(X - X^*) = 0 \qquad \forall \ell \in L \text{ (p-formes linéaires en } X - X^*_s)$$

$$C'_j(X^*)(X - X^*) \geq 0 \qquad \forall j \in J \text{ (q-formes linéaires en } X - X^*_s)$$

$$C_k(X^*) + C'_k(X^*)(X - X^*) > 0 \qquad \forall k \in K.$$

La dernière inégalité sera toujours vérifiée si $X - X^*$ est assez petit ; dans ce cas les contraintes unilatérales vérifiées non saturées n'interviennent pas dans l'étude d'un maximum local. En appliquant le théorème précédent, le problème peut être résumé comme suit :

Trouver un tableau X^* tel que :

$$f'(X^*)(X - X^*) = \sum_{\ell \in L} \lambda_\ell C'_\ell(X^*) \cdot (X - X^*) + \sum_{j \in J} \lambda_j C'_j(X^*) \cdot (X - X^*)$$

et

$$\lambda_j \leq 0 \qquad \forall j \in J.$$

La relation précédente peut être vérifiée pour tout X appartenant au domaine admissible, si :

$$f'(X^*) = \sum_{\ell \in L} \lambda_\ell C'_\ell(X^*) + \sum_{j \in J} \lambda_j C'_j(X^*) \qquad (4)$$

A l'optimum λ_ℓ et λ_j portent les noms respectifs de multiplicateurs de Lagrange et de multiplicateurs de Kuhn et Tucker.

I - 3 TECHNIQUE DU GRADIENT PROJETE

La technique du gradient projeté est une technique itérative.

I.3.1 Initialisation

Nous supposerons connu le tableau initial d'une itération. Ce tableau sera soit X^o, à la première itération, soit X^r, tableau terminal de l'itération r.

Par hypothèse nous supposerons que X^o a été choisi pour vérifier toutes les contraintes :

$$C_\ell(X^o) = 0 \qquad \forall \ell \in L$$

$$C_j(X^o) \geq 0 \qquad \forall j \in J$$

Par construction tout terminal X^r vérifie ces mêmes contraintes.

I.3.2 Itérations :

Chaque itération comporte deux étapes :

- choix d'une direction de montée ;
- recherche d'une nouvelle solution.

i) Choix d'une direction de montée locale admissible :

lère phase : Le calcul des coefficients de Lagrange, Kuhn et Tucker :

Une fois le tableau initial, X^o ou X^r, déterminé, il s'agit d'abord de trouver une direction D admissible telle que :

$f(X^r + h\,.\,D)$ soit maximum pour $h > 0$, et $\|h\,.\,D\|$ donné.

Or, nous cherchons un maximum local, et en choisissant h petit, on peut linéariser la fonction $f(X^r + h\,.\,D)$ et les contraintes.

Ce qui permet d'écrire :

$$\left.\begin{aligned} f(X^r + h.D) &\sim f(X^r) + h.D\, f'(X^r) = f(X^r) + h.\sum_{s=1}^{n} d_s \frac{\partial f}{\partial x_s} \\ C_\ell(X^r + h.D) &\sim C_\ell(X^r) + h.D\; C'_\ell(X^r) \\ C_j(X^r + h.D) &\sim C_j(X^r) + h.D\; C'_j(X^r) \end{aligned}\right\} = C_i(X^r) + h\,.\sum_{s=1}^{n} d_s \frac{\partial C_i}{\partial x_s} \qquad (5)$$

$\forall i \in \{L \cup J\}$.

Soit D une direction admissible ; elle doit vérifier :

compte tenu des simplifications obtenues par linéarisation (relation 5), le problème devient :

maximiser $\quad \Omega = D\,.\,f'(X^*)$

avec les contraintes :

$$\|h\,.\,D\| = \delta$$

qui devient, pour h fixé :

$$\Psi_0 = \sum_{s=1}^{n} d_s^2 - \delta^2 = 0$$

où les $(d_1, d_2, \ldots, d_n)$ sont les élèments de D,

$$\Psi_\ell = D\,.\,C'_\ell(X^r) = 0 \qquad \forall \ell \in L$$

$$\Psi_j = D\,.\,C'_j(X^r) \geq 0 \qquad \forall j \in J$$

Rappelons que les inconnues de ce problème auxiliaire d'optimisation sont les éléments $d_1, d_2, \ldots, d_n$ du tableau D.

Supposons que les contraintes qui étaient saturées en X^r, restent saturées, alors en appliquant la formule (4) :

$$\Omega' = \lambda_o \Psi'_o + \sum_{\ell \in L} \lambda_\ell \Psi'_\ell + \sum_{j \in J} \lambda_j \Psi'_j \qquad (6)$$

où $\Omega' = \begin{bmatrix} \frac{\partial \Omega}{\partial d_1} = \frac{\partial f}{\partial x_1}(X^r) \\ \frac{\partial \Omega}{\partial d_2} = \frac{\partial f}{\partial x_2}(X^r) \\ \vdots \\ \frac{\partial \Omega}{\partial d_n} = \frac{\partial f}{\partial x_n}(X^r) \end{bmatrix}$; $\Psi'_{\ell \text{ ou } j} = \begin{bmatrix} \frac{\partial \Psi}{\partial d_1} = d_1 \frac{\partial \Psi}{\partial x_1} \\ \frac{\partial \Psi}{\partial d_2} = d_2 \frac{\partial \Psi}{\partial x_2} \\ \vdots \\ \frac{\partial \Psi}{\partial d_n} = d_n \frac{\partial \Psi}{\partial x_n} \end{bmatrix}$ et $\Psi'_0 = 2\,.\,D$

et la relation (6) devient alors

$$f'(X^r) + 2\lambda_o D + \sum_{\ell \in L} \lambda_\ell C'_\ell(X^r) + \sum_{j \in J} \lambda_j C'_j(X^r).$$

En posant $2\lambda_o = 1$,

alors $\qquad D = f'(X^r) - \Sigma \lambda_i C'_i(X^r) \qquad \forall i \in \{L \cup J\} \qquad (7)$

D doit aussi vérifier les (p + q) équations :

$$D\,.\,C'_i(X^r) = 0 \qquad \forall i \in \{L \cup J\} \qquad (8)$$

Dès (7) et (8) :

$$[\Lambda] = [B^{-1}]\,[E] \qquad (9)$$

où $[\Lambda] = [\lambda_i] \qquad \forall i \in \{L \cup J\}$;

$[B] = [C']\,{}^{t}[C']$ matrice carrée symétrique de dimension $(p+q)\times(p+q)$ et $[E]$ est une matrice colonne d'élément générique :

$$e_i = \sum_{s=1}^{n} \frac{\partial f(X^r)}{\partial x_s} \cdot \frac{\partial C_i(X^r)}{\partial x_s}.$$

Le passage à la deuxième phase de la première étape dépend des valeurs de λ_i (coefficients de Kuhn et Tucker). Si tous les λ_i sont négatifs ou nuls, ou n'existent pas, on calcule la direction locale de meilleure montée D en utilisant l'équation (7). La direction D doit maintenir saturées les contraintes unilatérales qui étaient saturées en son origine X^r. Pour qu'il en soit ainsi il faut et il suffit que tous les coefficients de Kuhn et Tucker restent négatifs ou nuls (cf. [15]).

En examinant les valeurs absolues des d_i, $i \in J$ deux cas peuvent se présenter :

1) Cas d'optimalité, si :

$$|d_j| = 0 \qquad \forall j \in J.$$

Pour le cas de traitement sur ordinateur, cette condition est difficile à réaliser, et on considère que l'optimum est obtenu si $|d_j| \leq \varepsilon$, où ε est un paramètre arbitraire.

2) Si une ou plusieurs valeurs de $|d_j|$ sont différentes de zéro, on a intérêt à garder la direction D comme une direction de montée et on effectue la deuxième étape du calcul dans cette itération.

Revenons à la relation (9), et supposons que un au moins des coefficients de Kuhn et Tucker soit positif. Cela signifie qu'il y a des directions de montée locale plus satisfaisantes que D. Dans ce cas, nous chercherons une direction de montée admissible en ne libérant qu'une contrainte correspondant à un λ_i-positif.

ii) 2ème phase :

Cas où une contrainte est libérée

Soit $\lambda_{j_1} > 0$, la nouvelle direction locale de montée $\bar{D}$ doit vérifier :

$$\left.\begin{aligned} &\bar{D} = f'(X^r) - \Sigma \bar{\lambda}_i\, C_i'(X^r) ; \\ &\bar{D} \cdot C_i'(X^r) = 0 \\ \text{et} \quad &C_{j_1}(X^r + h \cdot \bar{D}) \geq 0 \end{aligned}\right\} \quad \forall i \in \{L \cup J'\}, \quad J' = \{J/j_1\}$$

les $\bar{\lambda}_i$ sont solutions de :

$$[\bar{\Lambda}] = [\bar{B}^{-1}]\,[\bar{E}] \qquad \text{où}$$

$[\bar{B}]$ et $[\bar{E}]$ sont les deux matrices définies par la relation (9) après la suppression dans la matrice C' de la ligne :

$$\frac{\partial C_{j_1}}{\partial x_1} \quad \frac{\partial C_{j_1}}{\partial x_2} \quad \cdots \quad \frac{\partial C_{j_1}}{\partial x_n} \ ;$$

et dans la matrice $[E]$ de l'élément e_{j_1}.

Après avoir libéré la contrainte C_{j_1} correspondant à $\lambda_{j_1} > 0$, on diminue le nombre des contraintes "utiles" par une, et on libère une variable de base.

Rappelons que [15] :

- D est la direction de meilleure montée locale lorsque tous les coefficients λ_i sont négatifs ou nuls ;
- D reste direction de montée si l'un ou plusieurs des λ_j sont positifs ;
- D reste direction de montée admissible si nous libérons une (et une seule) contrainte dont le coefficient de Kuhn et Tucker est positif.

I.3.3 Recherche d'une nouvelle solution réalisable X^{r+1} sur la direction de montée :

1ère phase : Recherche d'un tableau Y "meilleur" que X^r :

Le but de cette phase est de faire calculer un tableau Y (qui ne sera pas nécessairement la nouvelle solution X^{r+1}) réalisable et meilleur que X^r. Etant donnée une direction de montée locale D, et un pas h strictement positif, on cherche le tableau :

$$Y = X^r + h \cdot D$$

qui doit vérifier :

$$\left.\begin{array}{ll} C_\ell (Y) = 0 & \forall \ell \in L \\ C_j (Y) = 0 & \forall j \in J'_r \\ C_k (Y) > 0 & \forall k \in K'_r \end{array}\right\} \qquad (10)$$

où J'_r est l'ensemble des indices des contraintes unilatérales saturées, K'_r est l'ensemble des indices des contraintes vérifiées, non saturées, au cours de cette étape, et

$$f(Y) > f(X^r) \qquad (11)$$

Si Y vérifie les conditions (10) et (11), alors Y est un tableau admissible et on passe à la deuxième phase de cette étape. Dans le cas où Y ne vérifie pas la condition (11), on divise h par 4 et on recommence le calcul de Y jusqu'à ce qu'on arrive à un tableau Y réalisable, qui satisfait (11) ; si h devient trop petit on arrête le calcul (il y a "incident").

Dans le cas où Y ne vérifie pas toutes les contraintes non saturées :

$$C_k (Y) > 0 \qquad k \in K'_r$$

on cherche par une technique de dichotomie deux valeurs pour h, h_1 et h_2 telles que :

a) $h_2 - h_1 \leq \varepsilon$ donné

b) $Y = X^r + h_1 \cdot D$ vérifie toutes les contraintes

c) il reste une et une seule contrainte non vérifiée par :

$$Y = X^r + h_2 \cdot D.$$

Après avoir trouvé les valeurs des h_1 et h_2 qui vérifient les conditions précédentes, on sature la contrainte non vérifiée, on fait entrer une variable hors base dans la base, et enfin on calcule X^{r+1} de façon à ce qu'il vérifie les nouvelles contraintes ainsi que l'inégalité (11).

2ème phase : Calcul de X^{r+1} avec détermination automatique du pas :

On considère que la seule variable à déterminer pour maximiser la fonction $f(X^r + h \cdot D)$ est la variable h. Pour cela, on approche cette fonction par un polynôme g (h) du second degré en h, et on cherche h^* qui maximise ce polynôme. Le polynôme g (h) peut être obtenu par l'interpolation sur le support $[0, \ell, 2\ell]$. Posons :

$$g_0 = f(X^r)$$

$$g_1 = f(X^r + h \cdot D)$$

$$g_2 = f(X^r + 2h \cdot D)$$

le polynôme sera :

$$g(h) = g_0 + \frac{h}{\ell}\left[-\frac{3}{2} g_0 + 2 g_1 - \frac{1}{2} g_2\right] + \frac{h^2}{2\ell^2}(g_0 - 2 g_1 + g_2)$$

et il possèdera un maximum si

$$g_0 - 2 g_1 + g_2 < 0 \qquad (12)$$

Ce maximum sera obtenu pour :

$$h^* = \frac{-\ell}{2} \cdot \frac{-3 g_0 + 4 g_1 - g_2}{g_0 - 2 g_1 + g_2} \qquad (13)$$

Pour cela, dans la première phase de cette étape, après le calcul du tableau Y qui est considéré comme un tableau meilleur que X^r avec un pas correspondant à h_1, on calcule le tableau Z de pas $2\,h_1$. En considérant que ce tableau est admissible et

$f(Z) \geq f(Y)$ (sinon, on prend $X^{r+1} = Y$ et le pas de l'itération suivante h_1) nous posons :

$$g_0 = f(X^r) \; ; \; g_1 = f(Y) \; ; \; g_2 = f(Z)$$

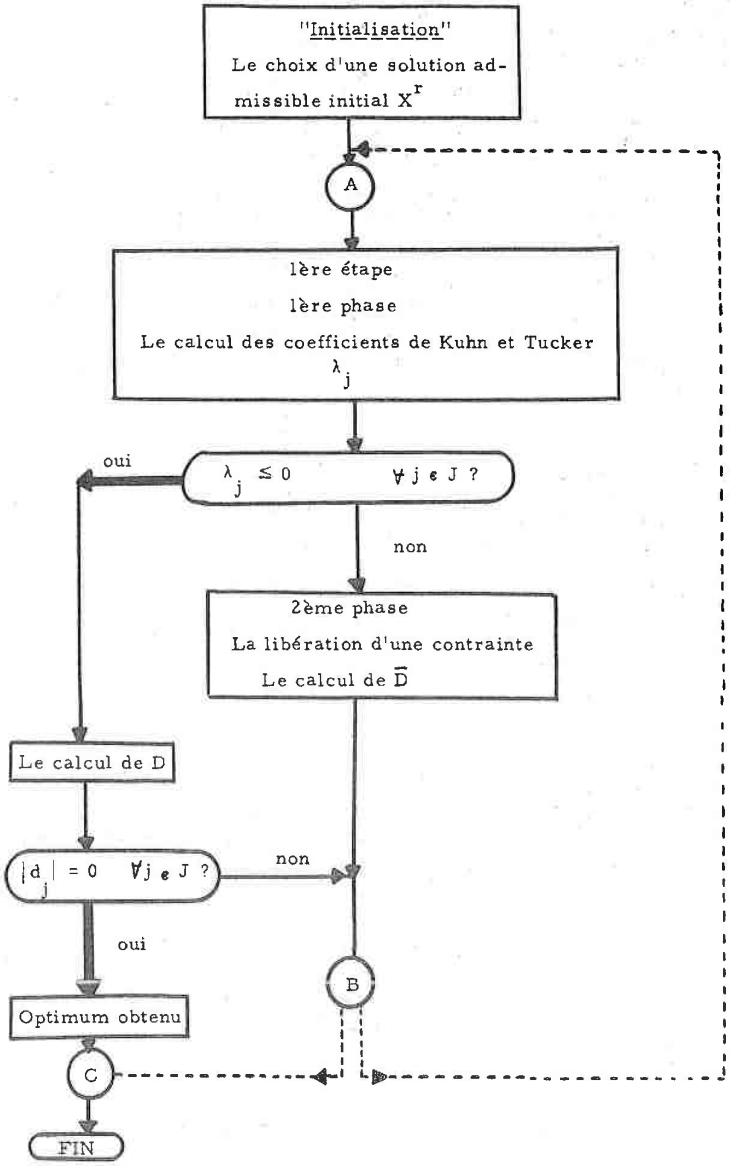
et nous examinons la condition (12). Si cette condition est satisfaite, on calcule le tableau U de pas h^* calculé dans (13).

Si U est calculable et admissible, nous prenons pour X^{r+1} le meilleur des trois tableaux Y, Z ou U, c'est-à-dire celui pour lequel :

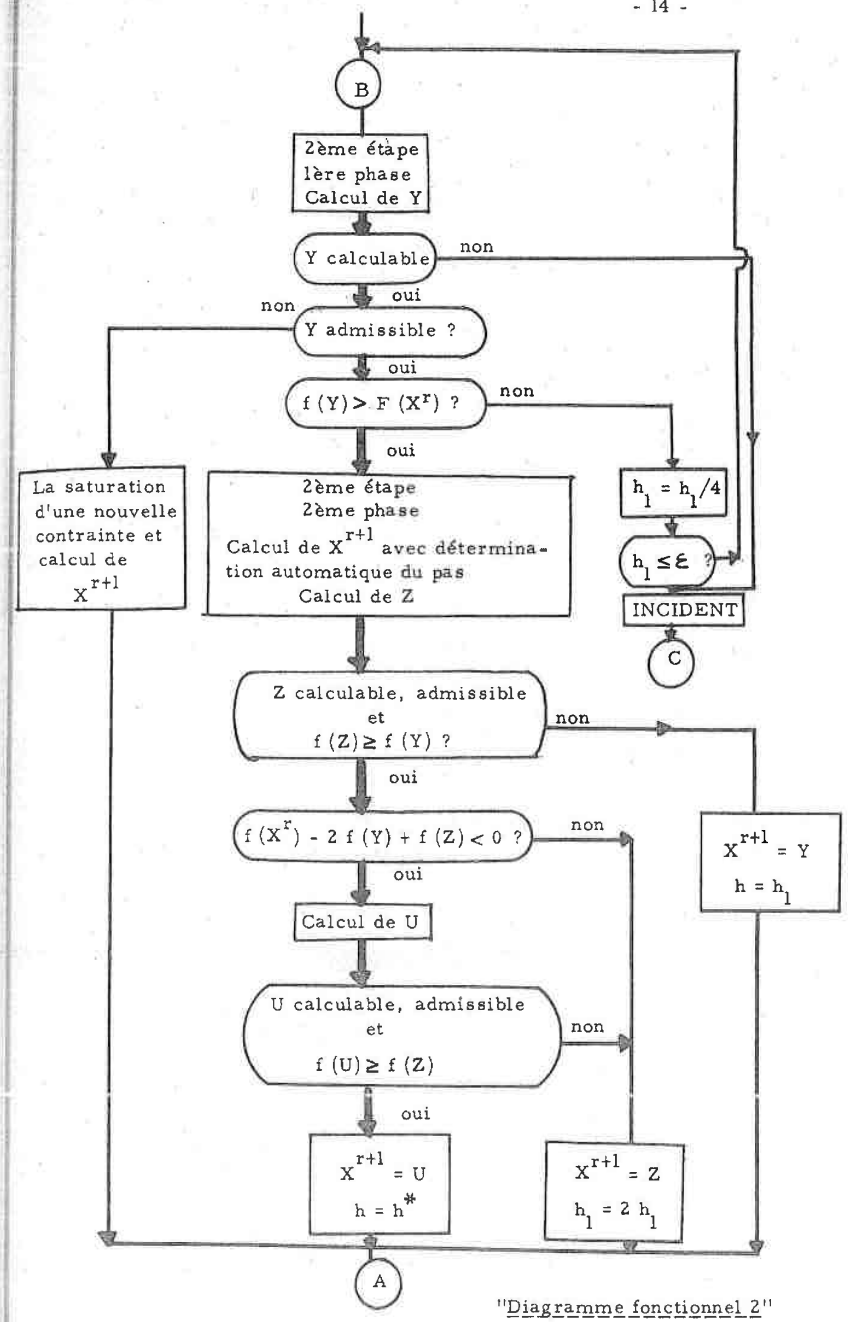
$$f(X^{r+1}) = \text{Sup}\,[\, f(Y),\ f(Z),\ f(U)\,].$$

Après le calcul de X^{r+1} et de son pas correspondant, on recommence le calcul à partir de la première phase de la deuxième étape.

Le processus du gradient projeté sera illustré par les deux organigrammes suivants.



"Diagramme fonctionnel 1"



"Diagramme fonctionnel 2"

I - 4 LES SOUS-PROGRAMMES DE BIBLIOTHEQUE UTILISES DANS LA METHODE DU GRADIENT PROJETE

Le but de notre recherche se limite à l'emploi de la méthode du gradient projeté pour la résolution des problèmes intervenant dans les problèmes de flots et de transports, problèmes où les contraintes vérifient des conditions linéaires spécifiques - particulièrement simple. Il sera donc nécessaire de simplifier certains sous-programmes de la bibliothèque d'optimisation [16] . Nous commencerons par décrire ces sous-programmes et le rôle de chacun avant de présenter les simplifications introduites.

- Le sous-programme GPRØJ (Gradient projeté) :

Il fait le calcul de la première étape de la méthode, c'est-à-dire qu'il cherche la meilleure direction de montée locale admissible D ou $\bar{D}$ selon les signes des coefficients de Kuhn et Tucker. Dans le cas ou on calcule $\bar{D}$, le sous-programme libère la contrainte correspondant à λ_{j_1} le plus grand, libère aussi une variable de base acceptable et enfin diminue le nombre de variables de base de 1.

- Le sous-programme MCADIR :

Il a deux rôles :

1 - il est indispensable pour calculer les coefficients de Lagrange de Kuhn et Tucker, d'où la nécessité de résoudre un système linéaire surdéterminé tel le système (4) :

$$^{t}[\, C'(X^{*}) \,]\ [\, \Lambda \,] = [\, f'(X^{*}) \,]$$

où la matrice $C'(X^{*})$ est rectangulaire de m-lignes, n-colonnes et $m \geq n$;

$f'(X^*)$ est une matrice colonne de m-éléments ;

et Δ est une matrice colonne de n-éléments.

2 - Il cherche la solution du système ordinaire :

$$[A][X] = [B]$$

où la matrice A est carrée.

La méthode de calcul appliquée est celle de GOLUB (multiplication par des matrices de Householder qui diagonalisent la matrice $C'(X^*)$ et minimisent la norme euclidienne de la matrice des résidus [14] :

$$[R] = {}^t[C'(X^*)][\Delta] - [f'(X^*)]$$

ou $$[R] = [A][X] - [B].$$

- Le sous-programme ETAP2 :

Ce sous-programme effectue le calcul de la deuxième étape de la technique en cherchant sur la direction de montée D ou $\bar{D}$ un tableau admissible Y, Z ou U meilleur que X^r.

Dans le cas ou Y n'est pas admissible, le sous-programme fait appel au sous-programme SATUR (saturation d'une contrainte) qui cherche par la méthode de dichotomie deux valeurs de h, h_1 et h_2 telles que :

- $X^r + h_1 . D$ vérifie toutes les contraintes ;
- $h_2 - h_1 \leq \varepsilon$ où ε est un paramètre donné ;
- il existe une seule contrainte non vérifiée par $X^r + h_2 . D$.

Le sous-programme sature cette contrainte, augmente le nombre des contraintes unilatérales saturées par un, fait entrer en base une variable libre; enfin il calcule le tableau X^{r+1} qui vérifie les nouvelles contraintes.

- Le sous-programme ELINUM (Elimination numérique)

Le vecteur X^r est un vecteur réalisable qui vérifie toutes les contraintes utiles du problème. Après le calcul de la direction de montée D, on a

intérêt à calculer un vecteur X^{r+1} réalisable et meilleur que X^r. Le vecteur X^{r+1} sera donné par la relation :

$$X^{r+1} = X^r + h . D$$

h est le pas donné ou calculé par le programme (cf. I-3-2).

Le rôle essentiel de ce sous-programme est de calculer les n-éléments du vecteur X^{r+1} qui vérifient les p-contraintes utiles ($p \leq n$) :

$$C(X_H^{r+1}, X_B^{r+1}) = 0 \qquad (14)$$

La méthode appliquée est la linéarisation (dûe à Newton). Le sous-programme comporte les étapes suivantes :

- la construction de la matrice carrée des dérivées des contraintes utiles par rapport aux variables de base $[C'_B]$ où :

$$[C'_B] = \begin{bmatrix} \partial C_1/\partial x_{b_1} & \partial C_1/\partial x_{b_2} & \cdots \\ \partial C_2/\partial x_{b_1} & \partial C_2/\partial x_{b_2} & \cdots \\ \vdots & & \\ \partial C_p/\partial x_{b_1} & \partial C_p/\partial x_{b_2} & \cdots \end{bmatrix}$$

- Or, les contraintes du problème sont non-linéaires, on calcule par une méthode itérative les vecteurs $X_B^0, X_B^1, \ldots, X_B^s, X_B^{s+1}$ où X_B^0 est arbitraire et X_B^{s+1} est défini en fonction de X^s comme solution du système linéaire :

$$[C'_B(X_H, X_B^s)][X_B^{s+1} - X_B^s] = \begin{bmatrix} C_1(X_H, X_B^s) \\ C_2(X_H, X_B^s) \\ \vdots \\ C_p(X_H, X_B^s) \end{bmatrix}$$

La fin des itérations sera obtenue lorsque :

$|C_i(X_H, X_B^S)| \leq \varepsilon$ (paramètre du sous-programme).

En plus du programme principal qui contient les paramètres et les données (cf. II.1) du problème, nous avons besoin de sous-programmes qui décrivent la fonction objectif à optimiser, sa dérivée par rapport aux variables, la matrice de toutes les contraintes, et enfin la matrice des dérivées partielles des contraintes par rapport aux variables. Ces sous-programmes portent les noms successifs :

FUNCTION F(X) ; GRADS ; CONT et DEPCO,

et doivent être écrits par l'utilisateur pour chaque application.

I - 5 APPLICATIONS AUX PROBLEMES DE TRANSPORT ET DE FLOT DANS UN GRAPHE

Nous nous proposons d'adapter, en les simplifiant, les méthodes générales précédemment exposées dans le cas de deux familles de problèmes particuliers :

le problème de transport et celui des flots dans un graphe.

Ces problèmes sont classiques lorsque la fonction objectif est linéaire (cf. [2, 8, 10, 12]), mais nous nous proposons de les étendre à des fonctions objectif non linéaires, la spécifité du problème traité venant alors des caractéristiques des contraintes, qui se traduisent en fait par le caractère totalement unimodulaire des matrices des contraintes.

Nous considérons pour chaque problème :

- une fonction objectif non linéaire,
- un ensemble de contraintes linéaires de caractères spécifiques, développés ultérieurement.

Dans la méthode décrite, il est nécessaire de construire et d'étudier la matrice des dérivées des contraintes "utiles" de chaque problème.

I.5.1 Problèmes de transport :

Il s'agit de minimiser le coût du transport entre N points de distribution P_i et M points de consommation R_j afin de satisfaire, toutes les demandes aux points R_j. La quantité x transportée de P_ℓ vers R_k sera notée par $x_{\ell,k}$, et le coût du transport de cette quantité s'exprime par la fonction non linéaire $g_{\ell,k}(x_{\ell,k})$.

Nous considérerons le problème dans les deux cas suivants :

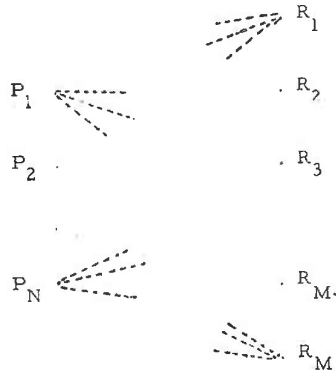
1 - la somme des quantités disponibles q_i aux points de distribution P_i est égale exactement à la somme des quantités demandées Q_j aux points R_j. C'est-à-dire que :

$$\sum_{i=1}^{N} q_i = \sum_{j=1}^{M} Q_j .$$

2 - La somme des quantités disponibles q_i aux points de distribution P_i dépasse les besoins aux points de consommation R_j, ou

$$\sum_{i=1}^{N} q_i > \sum_{j=1}^{M} Q_j .$$

Le problème peut être représenté par un graphe simple (fig. 1) dont les sommets sont soit les points de distribution P_i, soit les points de consommation R_j, et



(fig. 1)

dont l'arc $P_i R_j$ représente le chemin entre les sommets P_i et R_j.

Dans les deux cas précédents, les inconnues du problème sont les quantités transportées x_{ij} ($i = 1, 2, \ldots, N$; $j = 1, 2, \ldots M$) qui minimisent la fonction non linéaire :

$$F = \sum_{i=1}^{N} \sum_{j=1}^{M} g_{ij}(x_{ij})$$

D'autre part, le problème comporte des contraintes linéaires différentes selon le cas à traiter. Par la suite, on présente les contraintes que doit satisfaire chacun des deux cas.

I.5.1.a Le premier cas ($\sum_{i=1}^{N} q_i = \sum_{j=1}^{M} Q_j$) :

- Les contraintes bilatérales

$$\left.\begin{aligned} &\sum_{j=1}^{M} x_{ij} = q_i \qquad \text{pour } i = 1, 2, \ldots, N \\ &\sum_{i=1}^{N} x_{ij} = Q_j \qquad \text{pour } j = 1, 2, \ldots, M \end{aligned}\right\} \quad (15\text{ a})$$

- Les contraintes unilatérales :

$$x_{ij} \geq 0 \qquad \forall i, j \qquad (15\text{ b})$$

$\Big\}$ (15)

I.5.1.b Le deuxième cas ($\sum_{i=1}^{N} q_i > \sum_{j=1}^{M} Q_j$) :

Dans ce cas, on considère que :

- les contraintes bilatérales seront :

$$\sum_{i=1}^{N} x_{ij} = Q_j \qquad j = 1, 2, \ldots, M \qquad (15'\text{ a})$$

- Les contraintes unilatérales :

$$\left.\begin{aligned} &\sum_{j=1}^{M} x_{ij} \leq q_i \qquad i = 1, 2, \ldots, N \\ \text{ou} \quad &q_i - \sum_{j=1}^{M} x_{ij} \geq 0 \quad i = 1, 2, \ldots, N \\ \text{et} \quad &x_{ij} \geq 0 \qquad \forall i, j \end{aligned}\right\} \quad (15'\text{ b})$$

$\Big\}$ (15')

I.5.2 Caractéristiques de la matrice des dérivées des contraintes d'un problème de transport :

Nous prenons comme inconnues les M.N variables $x_1, x_2, \ldots, x_{MN}$ qui vérifient le système des contraintes (15), rangées dans un tableau tel le suivant :

x_1	x_2	x_3		x_M	q_1
x_{M+1}	x_{M+2}	x_{M+3}		x_{2M}	q_2
⋮	⋮	⋮		⋮	⋮
$x_{(N-1)M+1}$	$x_{(N-1)M+2}$			x_{MN}	q_N
Q_1	Q_2	Q_3		Q_M	

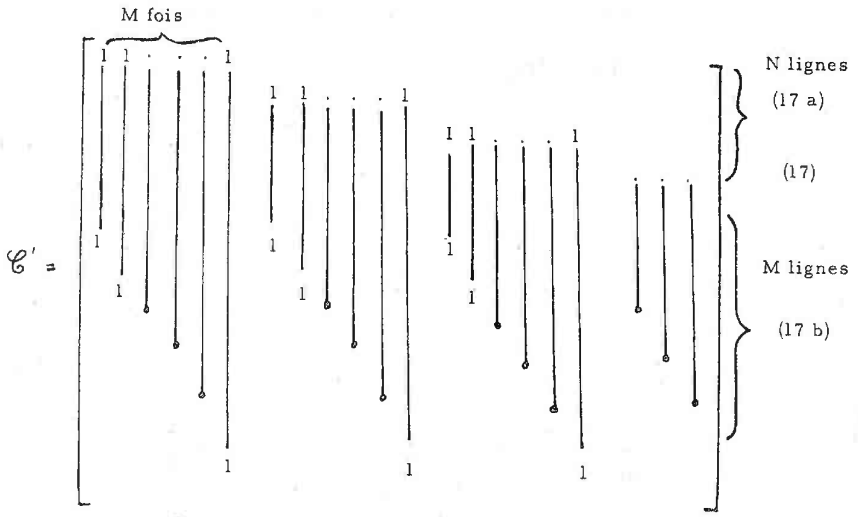
Le système des (M+N) contraintes bilatérales du système (15 a) peut être représenté par :

$$x_1 + x_2 + \ldots + x_M - q_1 = 0$$

$$x_{M+1} + x_{M+2} + \ldots + x_{2M} - q_2 = 0$$

$$
\begin{aligned}
&x_{(N-1)M+1} + x_{(N-1)M+2} + \ldots + x_{NM} - q_N = 0 \\
&x_1 + x_{M+1} + x_{2M+1} + \ldots + x_{(N-1)M+1} - Q_1 = 0 \qquad (16) \\
&x_2 + x_{M+2} + x_{2M+2} + \ldots + x_{(N-1)M+2} - Q_2 = 0 \\
&\text{- -} \\
&x_M + x_{2M} + x_{3M} + \ldots + x_{MN} - Q_M = 0
\end{aligned}
$$

La linéarité des contraintes : $C_\ell(X) \quad \forall \ell \in L$ fait que les dérivées $\partial C_\ell(X) / \partial x_k$; k = 1, 2, ..., MN se réduisent aux coefficients des inconnues x_k dans les contraintes. La matrice des dérivées du système des contraintes (16) a (N+M) lignes, M. N colonnes et prend la forme :



Il est clair que la matrice (17) :

- ne contient que des 0 et des 1,
- dans chacune des colonnes, il y a au plus deux coefficients non nuls.

Ceci traduit le fait que chaque variable x_k est représentée deux fois dans l'ensemble des contraintes (16), une fois dans les N premières équations et l'autre dans les M équations suivantes.

Aussi la matrice (17) peut elle être considérée comme une matrice d'incidence aux arcs du graphe de la figure (1). Chaque arc x_k (k = 1, 2, ..., MN) relie deux sommets s, t où :

$$s \in \{P_1, P_2, \ldots, P_N\} \text{ et } t \in \{R_1, R_2, \ldots, R_M\}.$$

I. 5. 3 Problèmes de flot dans un graphe :

Supposons un flot Φ fixe traversant l'entrée x_0 d'un graphe G (X, U) fini et sans boucle ; on cherche le flot $\varphi(u)$ passant à chaque arc u du graphe qui maximise la fonction :

$$F = \sum_{u \in U} g(u)\, \varphi(u)$$

en respectant :

i) les contraintes bilatérales :

$$\text{a)} \quad \sum_{u \in U\bar{x}} \varphi(u) - \sum_{u \in Ux^+} \varphi(u) = 0 \qquad x_i \neq x_n$$

où x_n est la sortie du graphe, $U\bar{x}$ l'ensemble des arcs incidents intérieurement à x, et Ux^+ celui des arcs incidents extérieurement à x.

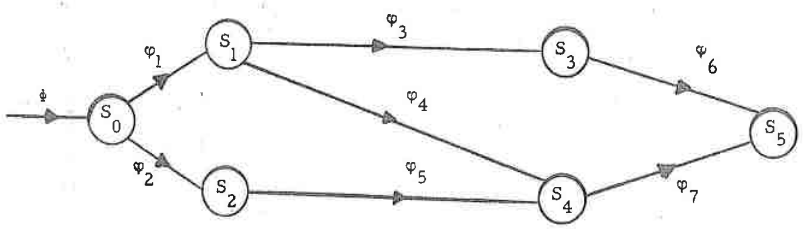
$$\text{b)} \quad \Phi - \sum_{u \in U x_o^+} \varphi(u) = 0$$

où x_o est l'entrée du graphe.

ii) Les contraintes unilatérales :

$$\left.\begin{aligned} &\text{c)} \quad p(u) - \varphi(u) \geq 0 \\ &\text{où } p(u) \text{ est la capacité de l'arc u.} \\ &\text{d)} \quad \varphi(u) \geq 0. \end{aligned}\right\} \qquad (17')$$

Considérons par exemple le graphe particulier suivant (fig. 2), en notant le flot traversant l'arc u_i par φ_i :



(fig. 2)

Le système des contraintes bilatérales est :

$$\begin{aligned}
\varphi_1 - \varphi_3 - \varphi_4 &= 0 \\
\varphi_2 - \varphi_5 &= 0 \\
\varphi_3 - \varphi_6 &= 0 \\
\varphi_4 + \varphi_5 - \varphi_7 &= 0 \\
\varphi_1 + \varphi_2 - \Phi &= 0.
\end{aligned} \qquad (18)$$

Le système des contraintes bilatérales ci-dessus traduit la "loi des nœuds" de Kirckhoff en électricité.

I. 5. 4 L'étude du comportement de la matrice des dérivées des contraintes du problème des flots dans un graphe :

En rappelant que les contraintes (18) sont linéaires, les coefficients des φ_i dans les contraintes peuvent être représentés par :

$$\mathcal{C}' = \frac{\partial C_\ell(\varphi)}{\partial \varphi_k} \qquad \forall \ell \in L \ ; \ k = 1, 2, \ldots, t$$

où t est le nombre des arcs dans le graphe.

Alors, la matrice $\mathcal{C}'$ des dérivées du système des contraintes (18) sera :

$$\left[\begin{array}{ccccccc}
1 & 0 & -1 & -1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & -1 \\
\hline
1 & 1 & 0 & 0 & 0 & 0 & 0
\end{array}\right] \begin{array}{l} \Big\} \text{(19 a)} \\ \\ \text{(19 b)} \end{array} \qquad (19)$$

En retenant la dernière ligne de la matrice (19), on peut constater sur le reste de la matrice que :

a) elle ne contient comme coefficients que des 0, 1 et -1 ;

b) chaque ligne traduit la conservation des flots aux sommets correspondant à cette ligne sauf pour les sommets entrée et sortie du graphe ;

c) chaque colonne possède au plus deux éléments non nuls, et les deux sont opposés.

Les deux matrices (17) et (19) ont en commun les propriétés suivantes :

- le coefficient a_{ij} de chaque matrice est égal soit à 0, soit à ± 1 ;
- chaque colonne contient au plus deux éléments non nuls ;
- le déterminant de chaque sous-matrice carrée extraite de la matrice (17) ou (19) est égal à 0 ou ± 1, comme nous allons le démontrer en I. 5. 5. a et I. 5. 5. b.

Ces propriétés sont caractéristiques des matrices totalement unimodulaires.

I. 5. 5. Matrices totalement unimodulaires

Définition 1 :

Soit une matrice $A = [\, a_{ij} \,]$ a n-lignes et p-colonnes dont chaque coefficient a_{ij} est un élément de l'ensemble $\{0, 1, -1\}$. A est dite totalement unimodulaire si le déterminant de toute sous-matrice carrée extraite de A appartient à cet ensemble d'entiers.

Définition 2 :

La sous matrice A' de A est dite Eulérienne si elle a un nombre pair non nul de coefficients non nuls dans chaque ligne et dans chaque colonne.

$$\forall i \in I' : \sum_{j \in J'} a'_{i,j} \equiv 0, \text{ mod } 2$$

et

$$\forall j \in J' : \sum_{i \in I'} a'_{i,j} \equiv 0, \text{ mod } 2$$

où $I' \subset I$; $J' \subset J$;

$I = \{1, 2, \ldots, n\}$ et $J = \{1, 2, \ldots, p\}$

Les propriétés des matrices totalement unimodulaires ont fait l'objet de nombreuses études (cf. [2, 3, 4, 5, 6, 7, 9, 10]).

Deux théorèmes importants concernent ce type de matrice, ce sont les suivants :

Théorème 1 de (Heller-Tompkins-Gale) [2] (p. 136)

Soit A une matrice de coefficients 0, 1 ou -1, telle que toute colonne contienne au plus deux coefficients non nuls, alors A est totalement unimodulaire si et seulement si ses lignes peuvent être réparties en deux ensembles disjoints I_1 et I_2, avec les deux conditions suivantes :

(1) si deux coefficients non nuls d'une même colonne ont le même signe, l'un est dans I_1 et l'autre dans I_2 ;

(2) si deux coefficients non nuls d'une même colonne ont des signes contraires, ils sont tous deux dans I_1 ou tous deux dans I_2.

Théorème 2 [4] :

A est une matrice totalement unimodulaire si et seulement si toute sous-matrice carrée Eulérienne est singulière.

I.5.5.a Cas du problème de transport

D'après le théorème de Heller-Tompkins et Gale, on peut démontrer qu'une matrice définie comme celle de (17) est une matrice totalement unimodulaire. Par exemple, pour un problème de transport avec M = 3 et N = 5, les (M + N-1) contraintes bilatérales peuvent être considérées comme :

$$x_1 + x_2 + x_3 + x_4 + x_5 - q_1 = 0$$
$$x_6 + x_7 + x_8 + x_9 + x_{10} - q_2 = 0$$
$$x_1 + x_6 + x_{11} - Q_1 = 0$$
$$x_2 + x_7 + x_{12} - Q_2 = 0$$
$$x_3 + x_8 + x_{13} - Q_3 = 0$$
$$x_4 + x_9 + x_{14} - Q_4 = 0$$
$$x_5 + x_{10} + x_{15} - Q_5 = 0$$

x_1	x_2	x_3	x_4	x_5	q_1
x_6	x_7	x_8	x_4	x_{10}	q_2
x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	q_3
Q_1	Q_2	Q_3	Q_4	Q_5	

et la matrice des dérivées des contraintes sera :

$$\mathcal{C}' = \left[\begin{array}{ccccccccccccccc}
1 & 1 & 1 & 1 & 1 & & & & & & & & & & \\
 & & & & & 1 & 1 & 1 & 1 & 1 & & & & & \\
\hline
1 & & & & & 1 & & & & & 1 & & & & \\
 & 1 & & & & & 1 & & & & & 1 & & & \\
 & & 1 & & & & & 1 & & & & & 1 & & \\
 & & & 1 & & & & & 1 & & & & & 1 & \\
 & & & & 1 & & & & & 1 & & & & & 1
\end{array}\right] \begin{array}{l} \} \text{ (20 a)} \\ \\ \} \text{ (20 b)} \end{array} \quad \text{(20)}$$

Il est clair que la matrice $\mathcal{C}'$ a les propriétés suivantes :

1) chaque coefficient C'_{ij} de la matrice $\mathcal{C}'$ appartient à l'ensemble {0, 1, -1}

2) chaque colonne de $\mathcal{C}'$ contient au plus deux éléments non nuls égaux à 1 ;

3) il est possible de répartir les lignes de $\mathcal{C}'$ en deux sous-ensembles I_1 et I_2 disjoints tels que :

- I_1 est l'ensemble des lignes correspondant à (20 a)
- I_2 est l'ensemble des lignes correspondant à (20 b).

Donc la matrice $\mathcal{C}'$ représentée par (20) est totalement unimodulaire.

En considérant l'exemple précédent dans l'autre cas du problème de transport (I.5.1.b), le système des contraintes (15') peut s'écrire sous la forme :

$$\begin{aligned} &x_1 + x_6 + x_{11} - Q_1 = 0 \\ &x_2 + x_7 + x_{12} - Q_2 = 0 \\ &\vdots \\ &x_5 + x_{10} + x_{15} - Q_5 = 0 \end{aligned}$$

$$\begin{aligned} &q_1 - (x_1 + x_2 + x_3 + x_4 + x_5) \geq 0 \\ &q_2 - (x_6 + x_7 + x_8 + x_9 + x_{10}) \geq 0 \\ &q_3 - (x_{11} + x_{12} + x_{13} + x_{14} + x_{15}) \geq 0 \end{aligned} \qquad (21)$$

et

$$x_i \geq 0 \qquad i = 1, 2, \ldots, 15.$$

Considérons d'abord la matrice des dérivées des contraintes bilatérales. Cette matrice prendra la forme :

$$\mathcal{C}' = \begin{bmatrix} 1 & & & & & 1 & & & & & 1 & & & & \\ & 1 & & & & & 1 & & & & & 1 & & & \\ & & 1 & & & & & 1 & & & & & 1 & & \\ & & & 1 & & & & & 1 & & & & & 1 & \\ & & & & 1 & & & & & 1 & & & & & 1 \end{bmatrix}$$

Il sera facile, en appliquant la définition 1, d'assurer qu'une matrice comme celle ci-dessus est totalement unimodulaire.

Au paragraphe I.9.1, nous démontrerons qu'une telle matrice restera totalement unimodulaire après la saturation d'une ou plusieurs des contraintes de signe.

Examinons maintenant la matrice ci-dessus après avoir saturé une ou plusieurs des contraintes de type (21). Considérons qu'après quelques itérations les contraintes (21) sont devenues saturées. La matrice $\mathcal{C}'$ prendra la forme :

$$\mathcal{C}' = \begin{bmatrix} 1 & & & & & 1 & & & & & 1 & & & & \\ & 1 & & & & & 1 & & & & & 1 & & & \\ & & 1 & & & & & 1 & & & & & 1 & & \\ & & & 1 & & & & & 1 & & & & & 1 & \\ & & & & 1 & & & & & 1 & & & & & 1 \\ -1 & -1 & -1 & -1 & -1 & & & & & & & & & & \\ & & & & & -1 & -1 & -1 & -1 & -1 & & & & & \\ & & & & & & & & & & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

En appliquant le théorème 1, on peut constater que la matrice $\mathcal{C}'$ ci-dessus reste totalement unimodulaire après chaque saturation d'une contrainte du système (21).

I.5.5.b Cas de flot dans un graphe

La matrice $\mathcal{C}'$ représentée par (19) est aussi totalement unimodulaire car :

1) chaque coefficient $C'_{ij} \in \{0, 1, -1\} \quad \forall i \in I, \quad \forall j \in J$;

2) on peut répartir les lignes des $\mathcal{C}'$ en deux sous-ensembles I_1 et I_2 disjoints tels que :

- I_1 est l'ensemble des lignes représenté par (19 a), où dans chaque colonne il y a au plus deux éléments non nuls de signe contraire.

- I_2 est la ligne représentée par (19 b) et chacun de ces éléments appartient à l'ensemble $\{0, 1, -1\}$.

Alors la matrice $\mathcal{C}'$ représentée par (19), sera aussi totalement unimodulaire.

I - 6 SIMPLIFICATION DE QUELQUES SOUS-PROGRAMMES DE LA BIBLIOTHEQUE

Le fait que les contraintes soient linéaires et que la matrice des dérivées des contraintes $\mathcal{C}'$ dans les deux problèmes soit totalement unimodulaire, nous permet de simplifier les traitements d'élimination, ce qui entraîne la modification de certains sous-programmes de la bibliothèque.

I.6.1 La résolution d'un système d'équations linéaires

Pour la recherche sur la direction D d'un vecteur X^{r+1}

réalisable vérifiant toutes les contraintes utiles (non linéaires) fournies par le système (14), on applique dans le cas général la méthode d'élimination numérique (sous-programme ELINUM) décrite en (I-4).

Lorsque les contraintes sont linéaires, il suffit de résoudre un système linéaire de p-équations à p-inconnues. En considérant que le système des contraintes utiles à vérifier par le vecteur X^{r+1} est sous la forme :

$$C\,(X_H^{r+1},\ X_B^{r+1}) = 0$$

Ce système peut s'écrire sous forme matricielle comme

$$A \,.\, X_B^{r+1} + K.\, X_H^{r+1} = D \qquad (22)$$

où A est une matrice carrée d'ordre p représentée par :

$$A = \begin{bmatrix} \partial C_1/\partial x_{b_1} & \partial C_1/\partial x_{b_2} & \cdots & \partial C_1/\partial x_{b_p} \\ \partial C_2/\partial x_{b_2} & \partial C_2/\partial x_{b_2} & \cdots & \partial C_2/\partial x_{b_p} \\ \vdots & \vdots & & \vdots \\ \partial C_p/\partial x_{b_1} & \partial C_p/\partial x_{b_2} & \cdots & \partial C_p/\partial x_{b_p} \end{bmatrix}$$

et K la matrice d'ordre $p \times (n-p)$ représentée par :

$$K = \begin{bmatrix} \partial C_1/\partial x_{h_1} & \partial C_1/\partial x_{h_2} & \cdots & \partial C_1/\partial x_{h_{n-p}} \\ \partial C_2/\partial x_{h_1} & \partial C_2/\partial x_{h_2} & \cdots & \partial C_2/\partial x_{h_{n-p}} \\ \vdots & \vdots & & \vdots \\ \partial C_p/\partial x_{h_1} & \partial C_p/\partial x_{h_2} & \cdots & \partial C_p/\partial x_{h_{n-p}} \end{bmatrix}$$

I.6.2. Le cas d'une matrice unimodulaire

Les p-colonnes de la matrice A constituent alors p-colonnes de la matrice $\mathcal{C}'$ représentée par (17) ou (19), et les (n-p) colonnes de la matrice K sont les (n-p) colonnes qui restent de la matrice $\mathcal{C}'$. En supposant que la matrice A soit inversible, le système des variables de la base X_B^{r+1} qui vérifie le système linéaire (22) sera donné par :

$$X_B^{r+1} = A^{-1} (D - K X_H^{r+1}). \qquad (23)$$

Or, les coefficients de la matrice A appartiennent à l'ensemble {0, 1, -1}, on peut constater que les coefficients de A^{-1} appartiendront au même ensemble car :

$$A \cdot A^{-1} = I$$

d'où $\quad$ dét A × dét A^{-1} = dét I = 1

alors $\quad$ dét A^{-1} = 1/ dét A = $\pm$ 1

aussi $\quad$ A^{-1} a comme élément $a'_{i,j}$ où

$$a'_{i,j} = (-1)^{i+j} \frac{M_{i,j}}{\text{dét A}}$$

où $M_{i,j}$ est la matrice carrée formée avec les cofacteurs de la matrice transposée ${}^tA$, associée à son élément de la ième ligne et de la jième colonne. Comme ${}^tA$ est aussi une matrice totalement unimodulaire, alors :

$$a'_{i,j} = 0 \text{ ou } \pm 1.$$

Donc, chaque ligne de A^{-1} ne contient comme éléments que 0, 1 et -1. Le système (23) peut s'écrire sous la forme matricielle :

$$X_B^{r+1} = A^{-1} \cdot R \qquad (24)$$

où R est un vecteur d'ordre p, d'élément générique r_i :

$$r_i = d_i - \sum_{j=1}^{n-p} k_{i,j} \cdot x_{h_j} \qquad i = 1, 2, \ldots, p \qquad (25)$$

A partir de (24) également, les valeurs de X_B^{r+1} s'obtiennent par la relation

$$x_{b_i} = \sum_{j=1}^{p} a'_{i,j} \cdot r_j \qquad i = 1, 2, \ldots, p \qquad (26)$$

Il est clair que r_i calculé par (25) et x_{b_i} calculé par (26) résultent des seules opérations d'additions et soustractions (selon les signes de $k_{i,j}$ et $a'_{i,j}$).

Cette méthode nécessite le calcul de A^{-1}, qui apparait fastidieux et coûteux pour une matrice d'origine A qui ne contient que 0, 1 et -1 comme coefficients.

En fait, nous éviterons le calcul de A^{-1} et calculerons directement la solution X_B^{r+1} par élimination, où les calculs seront simplifiés par le caractère unimodulaire de A.

I.6.3 L'application de la méthode d'élimination complète :

Une méthode directe est celle de Jordan dite aussi méthode d'élimination complète [14] . En groupant les coefficients de la matrice A et le vecteur R du système (24) dans une matrice A' de dimension (p, p+1), les inconnues X peuvent être calculées par les seules opérations d'additions et de soustractions sur les lignes de cette matrice A' afin de transformer la matrice A qui ne contient que 0 ou $\pm$ 1 en une matrice unité.

Nous allons démontrer qu'après chaque opération de transformation, la matrice A reste totalement unimodulaire.

Proposition 1 :

Un système linéaire peut être transformé en un système linéaire équivalent par :

- la permutation de deux lignes,
- l'addition à une ligne des éléments correspondants d'une autre ligne,
- la multiplication d'une ligne par -1.

Or, la matrice A de départ est totalement unimodulaire, les éléments des lignes satisfont aux théorèmes (1) et (2). En prenant deux lignes de A, le déterminant de chaque sous-matrice carrée extraite de ces deux lignes sera égal à 0, 1 ou -1.

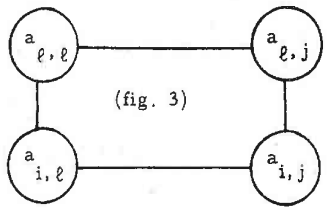
Nous voulons démontrer que l'addition de deux lignes ou la soustraction de deux lignes d'une matrice totalement unimodulaire, laisse la matrice totalement unimodulaire. Pour appliquer la méthode de Jordan, on cherche un élément (pivot) positif sur la diagonale principale de A, et on élimine les coefficients non nuls qui se trouvent dans la même colonne du pivot par l'addition ou la (soustraction) des deux lignes. La matrice A de départ est totalement unimodulaire. Supposons qu'elle reste totalement unimodulaire à l'étape ℓ de transformation. En considérant que le pivot $a_{\ell\ell} = 1$; deux cas possibles peuvent se présenter :

a) il y a un deuxième coefficient non nul (noté par $a_{i,\ell}$) dans la même colonne du pivot qui se trouve dans la ligne i et $a_{i,\ell} = 1$. Alors, on peut éliminer $a_{i,\ell}$ par la soustraction des coefficients de la ligne i à ceux de la ligne ℓ

b) Le deuxième coefficient non nul $a_{i,\ell}$ dans la même colonne du pivot se trouve dans la ligne i, et $a_{i,\ell} = -1$. C'est-à-dire qu'on peut éliminer $a_{i,\ell}$ par l'addition des coefficients de la ligne i à ceux de la ligne ℓ.

Un élément $a'_{i,j}$ dans la nouvelle ligne i peut être défini par la relation :

$$a'_{i,j} = a_{i,j} - \frac{a_{\ell,j} * a_{i,\ell}}{a_{\ell,\ell}} = a_{i,j} - a_{\ell,j} * a_{i,\ell} \qquad (27)$$



(fig. 3)

Après la transformation, $a_{i,\ell} = 0$. Il est clair que, si $a_{i,\ell} = 0$ avant la transformation, le coefficient $a_{i,j}$ ne changera plus.

Proposition 2 :

En considérant que les coefficients $a_{i,j}$; $a_{\ell,j}$ et $a_{i,\ell}$ appartiennent à {0, 1, -1}, alors le coefficient :

$$a'_{i,j} = a_{i,j} - a_{\ell,j} * a_{i,\ell}$$

sera un élément du même ensemble si et seulement si le produit des trois coefficients $a_{i,j} * a_{\ell,j} * a_{i,\ell}$ est positif ou nul.

Démonstration :

Le produit :

$$a_{i,j} * a_{\ell,j} * a_{i,\ell} < 0 \implies a_{i,j} \quad a_{\ell,j} \quad a_{i,\ell}$$

-1	-1	-1	d'où $a'_{i,j}$	$= -2 \notin \{0,1,-1\}$
1	1	-1	"	$= 2$ "
1	-1	1	"	$= 2$ "
-1	1	1	"	$= -2$ "

alors

$$a'_{i,j} \in \{0, 1, -1\} \implies a_{i,j} * a_{\ell,j} * a_{i,\ell} \geq 0.$$

Réciproquement :

Si $a'_{i,j} \in \{0, 1, -1\}$ alors $a'_{i,j} = 0$ ou ± 1.

$a'_{i,j}=0 \iff a_{i,j} - a_{\ell,j} * a_{i,\ell}=0 \Rightarrow a_{i,j} = 0$ et $a_{\ell,j} * a_{i,\ell} = 0 \Rightarrow a_{i,j} * a_{\ell,j} * a_{i,\ell}=0$

$a_{i,j} = 1$ et $a_{\ell,j} * a_{i,\ell} = 1 \Rightarrow \ldots\ldots\ldots = 1$

$a_{i,j} = -1$ et $a_{\ell,j} * a_{i,\ell} = -1 \Rightarrow \ldots\ldots\ldots = 1$

$a'_{i,j}=1 \iff a_{i,j} - a_{\ell,j} * a_{i,\ell} = 1 \Rightarrow a_{i,j} = 0$ et $-a_{\ell,j} * a_{i,\ell} = 1 \Rightarrow \ldots\ldots\ldots = 0$

$a_{i,j} = 1$ et $-a_{\ell,j} * a_{i,\ell} = 0 \Rightarrow \ldots\ldots\ldots = 0$

$a'_{i,j} = -1 \iff a_{i,j} - a_{\ell,j} * a_{i,\ell} = -1 \Rightarrow a_{i,j} = 0$ et $a_{\ell,j} * a_{i,\ell} = 1 \Rightarrow \ldots\ldots\ldots = 0$

$a_{i,j} = -1$ et $a_{\ell,j} * a_{i,\ell} = 0 \Rightarrow \ldots\ldots\ldots = 0$

alors

$$a'_{i,j} \in \{0, 1, -1\} \Rightarrow a_{i,j} * a_{\ell,j} * a_{i,\ell} \geq 0.$$

Revenons à la fig. 3, en considérant le cas où $a_{i,\ell} = \pm 1$, la transformation (27) sera équivalente à :

$$a'_{i,j} = a_{i,j} \mp a_{\ell,j}$$

le signe est pris selon le coefficient $a_{i,\ell}$ qui se trouve dans la même colonne du pivot.

Proposition 3 :

Le produit des coefficients de chaque sous-matrice carrée A' extraite de deux lignes d'une matrice totalement unimodulaire A est positif ou nul car :

a) si un coefficient de A' est égal à zéro, alors le produit des quatre coefficients de A' est égal à zéro.

b) Si les quatre coefficients sont non nuls, alors les coefficients

positifs (ou négatifs) seront de nombre pair, et alors le produit des quatre coefficients sera positif. Sinon, le déterminant de cette sous-matrice A' sera égal à ± 2 ce qui contredit la définition 1 de I.5.5. Par exemple, les sous-matrices suivantes :

$$\begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} ; \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} ; \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} , \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

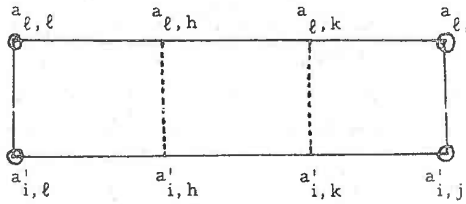
En suivant les propositions 2 et 3, les coefficients de la ligne transformée seront 0, 1 et -1.

Il nous reste à démontrer que le déterminant de chaque sous-matrice carrée A' extraite de la ligne du pivot et de la nouvelle ligne transformée reste toujours égal à 0 ou ± 1. Pour cela, deux cas possibles peuvent se présenter :

i) au moins un des quatre coefficients de A' est égal à zéro alors :

$$\text{Dét } (A') = 0 \text{ ou } \pm 1.$$

ii) Les quatre coefficients de A' sont non nuls (cas d'une matrice Eulérienne) ; c'est le cas quandles deux coefficients qui se trouvent sur la ligne du pivot sont non nuls, tandis que les deux autres coefficients avaient les valeurs zéro avant la transformation. Soit les quatre coefficients qui se trouvent sur les deux colonnes h, k (fig. 4) ; la relation (28) donne :



(fig 4)

$$\left.\begin{aligned} a'_{i,h} &= -a_{\ell,h} \\ a'_{i,k} &= -a_{\ell,k} \end{aligned}\right\} \text{ si } a_{i,\ell} = 1 \qquad (28)$$

$$\left.\begin{array}{l} a'_{i,h} = a_{\ell,h} \\ a'_{i,k} = a_{\ell,k} \end{array}\right\} \quad \text{si} \quad a_{i,\ell} = -1 \qquad (29)$$

Dans les deux cas (28) et (29) :

$$\text{Dét } (A') = 0.$$

Alors les coefficients de la nouvelle ligne transformée sont 0, 1 et -1 ; de plus le déterminant de chaque sous-matrice A' est égal à 0, 1 ou -1. C'est-à-dire que la matrice A restera une matrice totalement unimodulaire après transformation.

Revenons au système (24), en considérant que la matrice A est totalement unimodulaire ; il sera plus facile de trouver la solution (s'il y en a) de ce système par les seules opérations d'additions et de soustractions sur les lignes de A en appliquant la méthode d'élimination complète décrite en I.6.3. Cette méthode a été mise en application dans un sous-programme de bibliothèque qui porte le nom RESYS (cf. II-3).

I - 7 LA SATURATION D'UNE NOUVELLE CONTRAINTE

Lorsque nous avons obtenu le vecteur X^{r+1} par les techniques précédemment décrites, techniques qui n'ont pas fait intervenir les contraintes unilatérales vérifiées, mais non saturées en X^r, il reste à étudier si X^{r+1} vérifie ou non ces dernières contraintes. Deux cas peuvent se présenter :

1) toutes ces contraintes restent vérifiées, et nous enchaînons les itérations ;

2) une contrainte au moins devient saturée, dans ce cas,

les programmes de bibliothèque utilisent une méthode de dichotomie ; dans les problèmes particuliers que nous étudions, cette méthode est trop lourde et nous utiliserons une méthode mieux adaptée et plus rapide en cherchant l'intersection de la demi-droite décrite par X (avec $h > 0$) avec chacune des contraintes non saturées.

Rappelons que le vecteur :

$$X^{r+1} = X^r + h \,.\, D \qquad h > 0 \qquad (30)$$

doit vérifier :

- l'ensemble L des contraintes bilatérales :

$$C_\ell (X^{r+1}) = 0 \qquad \forall \ell \in L$$

- l'ensemble J' des contraintes unilatérales saturées :

$$C_j (X^{r+1}) = 0 \qquad \forall j \in J'_r \; ; \; J'_r \subset J$$

- l'ensemble K des contraintes unilatérales non saturées :

$$C_k (X^{r+1}) > 0 \qquad \forall k \in K \text{ où } K = \{J \,/\, J'_r\}.$$

Lorsque les contraintes sont linéaires, l'ensemble K peut s'écrire sous la forme matricielle :

$$A_i \,.\, X^{r+1} \geq b_i \qquad \forall i \in K.$$

Dès (30) :

$$A_i X^r + h \,.\, A_i D \geq b_i$$

d'où

$$h \,.\, A_i \,.\, D \geq b_i - A_i \,.\, X^r \qquad (31)$$

Puisque h est strictement positif, et X^r est un vecteur réalisable, donc :

$$b_i - A_i \,.\, X^r < 0.$$

Alors, parmi l'ensemble des contraintes unilatérales non saturées, on cherche l'ensemble des contraintes qui vérifie la condition :

$$A_i \cdot D < 0.$$

Soit S l'ensemble de telles contraintes où :

$$S = \{s \in K / A_i \cdot D < 0\}.$$

La condition (31) devient :

$$-h \cdot A_s \cdot D \leq A_s \cdot X^r - b_s \qquad \forall s \in S$$

d'où

$$h \leq \left(\frac{A_s \cdot X^r - b_s}{-A_s \cdot D}\right).$$

Au moins une contrainte de l'ensemble S sera saturée en posant :

$$h = \underset{s \in S}{\text{Min}} \left(\frac{A_s \cdot X^r - b_s}{-A_s \cdot D}\right).$$

La nature de la matrice des dérivées des contraintes permettra d'alléger le calcul des termes de la forme $A_s \cdot D$ par les seules opérations d'additions ou soustractions.

Cette méthode a remplacé la recherche par dichotomie dans le sous-programme SATUR de la bibliothèque. De plus, les deux sous-programmes de bibliothèque SATUR (après la modification) et ETAP2 sont réunis dans un sous-programme nommé ETAPL.

I. 7.1 Cas d'incident :

En examinant les contraintes des signes des deux problèmes de flot et de transport, on trouve qu'ils prennent la forme :

$$x_i \geq 0 \qquad (32)$$

Supposons que le vecteur X^r, de n-éléments est réalisable à l'itération r, la direction de montée associée est représentée par le vecteur

$D = d_i$ (i = 1, 2, . . . , n), et le vecteur X^{r+1} sera calculé par la relation (30).

Deux cas peuvent se présenter :

1) le vecteur X^{r+1} est réalisable et satisfait la condition

$$f(X^{r+1}) > f(X^r)$$

2) le vecteur X^{r+1} ne vérifie pas toutes les contraintes non saturées, d'où la nécessité de calculer le pas h' qui sature au moins une contrainte de l'ensemble {S}.

En effet, pour l'ensemble des contraintes unilatérales représenté par (32) :

$$\forall s \in \{S\} \quad x_s^{r+1} < 0 \implies x_s^r + h \cdot d_s < 0$$

Puisque x_s^r et h sont positifs, alors :

$$x_s^{r+1} < 0 \implies d_s < 0 \text{ et } |h \cdot d_s| > x_s^r$$

et la valeur de h' choisie sera :

$$h' = \underset{s \in S}{\text{Min}} \left(\frac{x_s^r}{-d_s}\right).$$

Pour des ordinateurs de faible précision, si x_s^r était trop petite, h' peut prendre la valeur zéro. Cela signifie que le nouveau vecteur X^{r+1} ne sera que le vecteur X^r, et on retombe à la même valeur de h'. Dans ce cas, on sature cette contrainte et on fait entrer dans la base variable x_s.

I - 8 CHANGEMENT AUTOMATIQUE DES VARIABLES DE BASE

Il sera nécessaire de changer le choix des variables de base chaque fois que l'ancienne n'arrive pas à résoudre un système comme (9) ou (14). Il

est facile de remplacer une liste de variables de base par une autre manuellement au début du calcul pour démarrer le programme, mais il sera difficile de faire ce changement après quelques itérations. Par exemple, dans un essai pour arriver à l'optimum d'un problème de transport, nous avons trouvé qu'il faut changer les variables de base une vingtaine de fois. Dans d'autres problèmes plus importants, le nombre de changements des variables de base sera peut être beaucoup plus grand, d'où la nécessité d'automatiser le changement des variables de base. Une technique est mise en œuvre par le sous-programme CRBAS (correction de base) pour choisir automatiquement la liste des variables de base.

L'algorithme commence par placer dans la base les variables qui interviennent seules dans certaines contraintes "utiles". Le reste de la liste des variables de base est choisi parmi les autres variables par une permutation circulaire.

En supposant que nous avons q' contraintes "utiles" et que le nombre des variables est n, on choisit parmi les n-variables les q' variables de base de façon à satisfaire aux deux conditions suivantes :

1) si une contrainte saturée ne dépend que d'une seule variable. Cette variable doit nécessairement être variable de base ;

2) chaque contrainte doit nécessairement faire intervenir au moins une variable de base.

La première condition est facile à réaliser en considérant deux tableaux de répèrage :

a) ILU : tableau de repérage des Indices des Liaisons Utilisées ; où le i-ème élément a la valeur 1 si la i-ème contrainte est bilatérale ou unilatérale saturée, la valeur 0 si la contrainte est unilatérale non saturée,

b) IVX : est un tableau défini comme suit :

IVX (I) = J, si la contrainte numéro I ne dépend que de X (J) ;

IVX (I) = 0, si la contrainte numéro I est quelconque.

Après avoir retenu les variables J correspondant aux ILU (i) = 1 et IVX (I) = J pour toutes les contraintes, on peut chercher le reste des variables en respectant la deuxième condition.

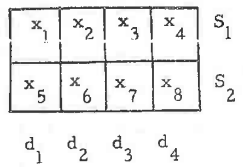
I.8.1 La première technique du changement des variables de base (CRBAS 1) :

Considérons que toutes les variables qui se trouvent dans les p-contraintes bilatérales sont rangées dans un anneau de façon que les p-premières variables (en choisissant la première variable dans chacune des p-contraintes bilatérales) soient rangées successivement suivies par les p-deuxième variables (en choisissant la deuxième variable dans chacune des p-contraintes bilatérales) et ainsi de suite.

Au début du calcul et après avoir satisfait la première condition, on cherche le reste des variables en commençant par la première position de l'anneau (fig. 4). A chaque changement on commence la recherche dans l'anneau à partir de la case correspondant au nombre des essais.

Par exemple, soit le système des contraintes bilatérales d'un problème de transport de deux sources et quatre destinations :

$$\left.\begin{aligned} x_1 + x_2 + x_3 + x_4 &= s_1 \\ x_5 + x_6 + x_7 + x_8 &= s_2 \\ x_1 + x_5 &= d_1 \\ x_2 + x_6 &= d_2 \\ x_3 + x_7 &= d_3 \end{aligned}\right\} \quad (34)$$

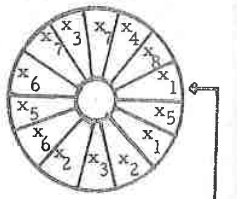


Les variables de base de la première itération seront :

$$x_1,\ x_5,\ x_2,\ x_3,\ x_6.$$

Si nous supposons que la première détermination a échoué, nous ferons un deuxième essai en commençant par la seconde case de l'anneau, et les variables de base seront :

$$x_5,\ x_1,\ x_2,\ x_3,\ x_6\ ;\ \text{etc.}$$



(fig. 4) le début de la 1ère itération

Si nous avons épuisé toutes les cases de l'anneau sans trouver une solution de base, on changera les contenus de l'anneau en supposant des changements dans les composantes de quelques contraintes. Par exemple, la deuxième contrainte peut être transformée en :

$$x_8 + x_6 + x_5 + x_7 = S_2\ ;$$

et la troisième contrainte en :

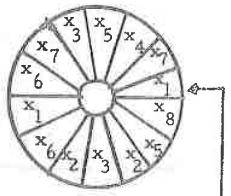
$$x_5 + x_1 = d_1.$$

Le contenu de l'anneau sera comme ci-contre (fig. 5). Les variables de base de la première itération seront :

$$x_1,\ x_8,\ x_5,\ x_2,\ x_3 \qquad (35)$$

et les variables de base de la deuxième itération seront :

$$x_8,\ x_5,\ x_2,\ x_3,\ x_6.$$



le début de la 1ère itération

(fig. 5)

Supposons que le changement de base s'effectue du point de départ jusqu'à l'arrivée en saturant la contrainte numéro I où : IVX (I) = 2 par exemple. En recommençant la recherche des variables de base à partir de la première case de l'anneau (fig. 5), les variables seront :

x_2 suivi par $x_1,\ x_8,\ x_5,\ x_3,\ x_6$, ce qui représentent une nouvelle base

différente de la base (35). Si la contrainte saturée I est du type : IVX (I) = 0, le sous-programme ETAPL choisit une variable hors base et la fait entrer dans la base ; on recommence alors la recherche des variables comme dans le cas précédent.

I.8.2 La deuxième technique du changement de base (CRBAS2)

Une autre méthode directe, mais qui nécessite un calcul plus compliqué et plus lourd, est de faire changer les variables de base par des "boucles" successives en nombre égal au nombre des contraintes bilatérales ; chaque boucle changeant un nombre égal de fois au nombre des variables qui se trouve dans la contrainte correspondant à cette boucle. Par exemple, pour le système (34), on peut changer les variables de base par les instructions :

```
DØ 1  I1 = 1, N1 où N1 : nombre de composantes de la 1ère contrainte
DØ 1  I2 = 1, N2 où N2 :   "        "        "   2ème    "
DØ 1  I3 = 1, N3 où N3 :   "        "        "   3ème    "
DØ 1  I4 = 1, N4 où N4 :   "        "        "   4ème    "
DØ 1  I5 = 1, N5 où N5 :   "        "        "   5ème    "
CALL CRBAS2.
```

Le sous-programme commence par examiner parmi les contraintes utiles, celles qui dépendent d'une seule variable X_J, c'est-à-dire les contraintes qui vérifient :

$$\text{ILU (I)} = 1,$$
$$\text{IVX (I)} = J$$

et la fait entrer dans la base. Or, dans notre cas les contraintes unilatérales sont de la forme (15 b) ou (17'), le nombre de ces variables est égal au nombre des contraintes unilatérales saturées, et en conséquence il reste à faire entrer

dans la base un nombre de variables égal au nombre des contraintes bilatérales. Ces dernières variables seront choisies à condition que :

- il existe au moins une variable de chaque contrainte bilatérale,
- la variable n'a pas encore figuré dans la liste des variables de base, c'est-à-dire que toutes les variables seront distinctes.

La première condition peut être vérifiée en retenant dans chaque contrainte bilatérale, la première variable libre rencontrée. Cela sera effectué en choisissant la première contrainte bilatérale par la technique précédente (la permutation circulaire) par rapport aux contraintes bilatérales et aux composantes de chacune. Après avoir choisi dans chaque contrainte bilatérale, une variable libre (si possible), et si le nombre de ces variables est inférieur au nombre des contraintes bilatérales, on recommence la recherche à partir de la première contrainte examinée.

Par exemple, dans le système (34), pour choisir les variables de base de la première itération, on introduit dans la base :

- x_1 de la première contrainte,
- x_5 de la deuxième contrainte,
- on saute la troisième contrainte, parce que les deux variables de cette contrainte ont déjà été choisis,
- x_2 de la quatrième contrainte,
- x_3 de la cinquième contrainte.

On recommence la recherche de la cinquième variable parmi les variables de la première contrainte en choisissant la première variable libre rencontrée (x_4) et la liste des variables de base sera :

$$x_1,\ x_5,\ x_2,\ x_3,\ x_4.$$

La recherche de la deuxième liste des variables de base sera commencée à partir de la deuxième contrainte et sera composée de :

- x_5 de la 2ème contrainte ;
- x_1 de la 3ème contrainte ;

- x_2 de la 4ème contrainte ;
- x_7 de la 5ème contrainte, et non x_3 à cause du changement de la dernière boucle ;
- x_3 de la 1ère contrainte,

et la liste des variables de base de la 2ème itération sera :

$$x_5,\ x_1,\ x_2,\ x_7,\ x_3.$$

Les listes des itérations qui suivent seront :

$$x_1,\ x_6,\ x_3,\ x_2,\ x_5$$

$$x_6,\ x_7,\ x_1,\ x_5,\ x_2$$

$$x_3,\ x_1,\ x_5,\ x_2,\ x_7$$

etc...

Le nombre des permutations possibles pour changer la liste des variables de base est $\prod_{i=1}^{P} N_i$, où P est le nombre des contraintes bilatérales. D'autre part, en appliquant cette méthode nous serons obligés d'introduire dans le programme principal les deux tableaux suivants :

1 - un vecteur colonne NVB de dimension P ; dans lequel se trouve le nombre des variables de chacune des P-contraintes.

2 - Un tableau entier LT de dimension $(P \times K)$, où $K = \text{Max } N_i$ $i = 1, 2, \ldots, P$. Ce tableau contiendra les variables des contraintes bilatérales différentes.

I - 9 LE CAS DE SATURATION OU LIBERATION D'UNE CONTRAINTE

Considérons dans les deux problèmes, les contraintes unilatérales qui dépendent d'une seule variable :

- les conditions de signe représentées par (15 b), dans le problème de transport ;

- les conditions de signe et les contraintes de capacité sur chaque arc représentées par (17'), dans le cas d'un problème de flot dans un graphe.

La saturation ou la libération d'une contrainte de ce type signifie l'addition ou la suppression d'une ligne et d'une colonne de la matrice des dérivées des contraintes par rapport aux variables de base. Cette ligne comporte un seul coefficient ± 1, correspondant à la variable entrante ou sortante.

I. 9.1 Cas de saturation d'une contrainte :

La matrice carrée $\mathcal{C}'_k$ de départ est totalement unimodulaire. Cette matrice est de dimension $(k \times k)$, où k est le nombre des contraintes bilatérales. Les coefficients dans la colonne i de cette matrice représentent les dérivées des contraintes bilatérales par rapport à la variable de base x_{b_i} choisie (fig. 6).

$$\mathcal{C}'_k = \begin{bmatrix} \partial C_1/\partial x_{b_1} & \partial C_1/\partial x_{b_2} & \cdots & \partial C_1/\partial x_{b_i} & \cdots & \partial C_1/\partial x_{b_k} \\ \partial C_2/\partial x_{b_1} & \partial C_2/\partial x_{b_2} & \cdots & \partial C_2/\partial x_{b_i} & \cdots & \partial C_2/\partial x_{b_k} \\ \vdots & & & & & \\ \partial C_k/\partial x_{b_1} & \partial C_k/\partial x_{b_2} & \cdots & \partial C_k/\partial x_{b_i} & \cdots & \partial C_k/\partial x_{b_k} \end{bmatrix} \quad \text{(fig. 6)}$$

où $\dfrac{\partial C_\ell}{\partial x_{b_j}} \in \{1, 0, -1\} \quad \forall \ell, j \in \{1, 2, \ldots, k\}$

Soit x_{b_ℓ} la variable entrant dans la base aprés avoir saturé la contrainte :

$x_\ell = 0 \qquad 1 \le \ell \le MN \qquad$ dans (15 b)

ou $\varphi(\ell) = 0 \qquad$ dans (17 ')

ou $\varphi(\ell) = p(\ell) \qquad$ dans (17').

La matrice $\mathcal{C}'_k$ devient de dimension (k + 1, k +1), le nombre des variables de base augmente de un, et $\mathcal{C}'_{k+1}$ prend la forme :

$$\mathcal{C}'_{k+1} = \left[\begin{array}{ccc:c:c} \partial C_1/\partial x_{b_1} & \partial C_1/\partial x_{b_2} \cdots & \partial C_1/\partial x_{b_\ell} & \cdots & \partial C_1/\partial x_{b_{k+1}} \\ \partial C_2/\partial x_{b_1} & \partial C_2/\partial x_{b_2} \cdots & \partial C_2/\partial x_{b_\ell} & \cdots & \partial C_2/\partial x_{b_{k+1}} \\ \vdots & & & & \\ \partial C_k/\partial x_{b_1} & \partial C_k/\partial x_{b_2} \cdots & \partial C_k/\partial x_{b_\ell} & \cdots & \partial C_k/\partial x_{b_{k+1}} \\ \hline 0 & 0 \cdots & \pm 1 & \cdots & 0 \end{array}\right] \begin{array}{l} \\ \\ \\ \\ \text{la contrainte } (k+1) \end{array}$$

ℓ-ième colonne

On peut constater que la matrice $\mathcal{C}'_{k+1}$ reste totalement unimodulaire car :

en considérant que la contrainte unilatérale saturée était du type : $x_\ell \ge 0$, alors le coefficient qui se trouve dans la ℓ-ième colonne et la (k+1) ième ligne sera + 1. Considérons maintenant qu'il y a un ou plusieurs coefficients dans la ℓ-ième colonne différent de zéro (fig. 7).

$$\mathcal{C}'_{k+1} = \left[\begin{array}{ccc:c:c} \partial C_i/\partial x_{b_1} & \partial C_i/\partial x_{b_2} \cdots & 1 & \cdots & \partial C_i/\partial x_{b_{k+1}} \\ \hline \partial C_j/\partial x_{b_1} & \partial C_j/\partial x_{b_2} \cdots & -1 & \cdots & \partial C_j/\partial x_{b_{k+1}} \\ \hline \partial C_k/\partial x_{b_1} & \partial C_k/\partial x_{b_2} \cdots & 1 & \cdots & \partial C_k/\partial x_{b_{k+1}} \\ \hline 0 & 0 & 1 & & 0 \end{array}\right] \quad \text{(fig. 7)}$$

Les coefficients de la matrice de départ $\mathcal{C}'_k$ forment une matrice totalement unimodulaire, c'est-à-dire que ces coefficients satisfont aux conditions du théorème 1. D'après la proposition 1, on peut transformer les coefficients qui se trouvent dans la ℓ-ième colonne en des coefficients qui vérifient le théorème 1 par des opérations d'addition ou de soustraction de la ligne (k + 1) aux autres lignes. Les autres coefficients de la matrice ne changent pas, puisque les autres coefficients de la (k + 1)-ième ligne sont des zéros. Aussi, il est clair que :

$$\text{Dét}\,(\mathcal{C}'_{k+1}) = \pm 1.\text{Dét}\,(\mathcal{C}'_k) = \pm\,\text{Dét}\,(\mathcal{C}'_k)$$

(en calculant le déterminant de la matrice $\mathcal{C}'_{k+1}$ par rapport aux coefficients de la dernière ligne).

I.9.2 Cas de libération d'une contrainte :

Nous avons démontré que la matrice $\mathcal{C}'_{k+1}$ des dérivées des contraintes par rapport aux variables de base reste totalement unimodulaire après la saturation d'une contrainte unilatérale. La libération d'une de ces contraintes peut arriver par la suppression d'une ligne et d'une colonne de la matrice $\mathcal{C}'_{k+1}$.

La suppression d'une ligne et d'une colonne d'une matrice totalement unimodulaire laisse les autres colonnes de la matrice vérifier les conditions du théorème 1, c'est-à-dire que la matrice reste totalement unimodulaire. C'est évident, puisque la matrice $\mathcal{C}'_{k+1}$ est déduite de la matrice $\mathcal{C}'_k$ qui était totalement unimodulaire.

I - 10 LE CHOIX INITIAL DES VARIABLES DE BASE :

Pour commencer le calcul dans chaque problème, nous avons besoin :

1 - d'un vecteur initial X^o,

2 - d'un choix initial des variables de base.

En ce qui concerne le premier point, le vecteur choisi X^o doit vérifier toutes les contraintes du problème, et le nombre des itérations pour atteindre l'optimum de la fonction objectif sera plus ou moins dépendant du choix de X^o.

L'importance du deuxième point apparaitra en cherchant un vecteur X^{r+1} meilleur que X^r. Pour cela, il faut que la matrice des dérivées des contraintes utiles par rapport aux variables de base $\mathcal{C}'_B$ soit inversible (cf. I.6.1). On tiendra alors compte des théorèmes et remarques classiques qui concernent chacun de ces problèmes pour choisir des variables de base convenables.

I.10.a Cas du problème de transport :

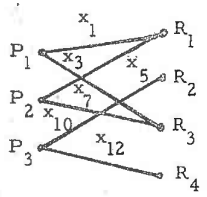
La solution de base dans le cas classique du problème de transport, est une solution qui comporte exactement (M - 1) (N - 1) valeurs nulles. Autrement dit une solution comportant :

$$MN - (M-1)(N-1) = M + N - 1 \quad \text{valeurs positives.}$$

Cette solution de base constitue un arbre dans le graphe non orienté correspondant aux arcs (P_i, R_j) seulement. Cet arbre contient M + N - 1 arêtes et entre deux sommets non adjacents, il existe une chaîne dont les sommets sont alternativement pris dans l'ensemble P_i et dans l'ensemble R_j (cf. I.5.1).

En considérant que chaque arête de l'arbre représente une variable de base, alors les variables (arêtes) choisies doivent vérifier (15 a) et (15 b). Par exemple, soit le tableau suivant :

	R_1	R_2	R_3	R_4	
P_1	x_1 5			x_4 10	15
P_2	x_5 10		x_7 5	x_8 5	20
P_3		x_{10} 10			10
	15	10	5	15	



(fig. 8)

La matrice $\mathcal{C}'_B$ ne sera pas inversible si nous prenons $x_1 = 5$; $x_3 = 0$; $x_5 = 10$; $x_7 = 5$; $x_{10} = 10$; $x_{12} = 0$, comme variables de base ; bien que les variables vérifient toutes les contraintes du problème. Ces variables ne constituent pas un arbre, mais possèdent plus d'une composante connexe.

I.10.b Cas du problème de flot dans un graphe :

Soit un graphe G (X, U) orienté fini et sans boucle, et un sous-ensemble de sommets $A \subset C$, on note par :

U_{A^+} : l'ensemble des arcs incidents intérieurement à A ;

U_{A^-} : l'ensemble des arcs incidents extérieurement à A.

Définition 1 [8] :

L'ensemble d'arcs U (A) est un cocycle et l'on a :

$$U(A) = U_{A^+} \cup U_{A^-}$$

et

$$\Phi = U_{A^+} \cap U_{A^-}$$

Définition 2 :

Un cocyle est élémentaire s'il est formé par l'ensemble des arcs qui relient entre eux deux sous-graphes connexes non vides, disjoints et dont la réunion constitue une composante connexe du graphe.

Considérons que le flux traversant chaque arc u_i du graphe G est borné par un couple de valeurs (a_i, b_i) réelles ou entières tel que $a_i \leq b_i$, le théorème suivant donne la condition pour qu'un flot puisse traverser le graphe.

Théorème [7] :

Soit G un graphe orienté et valué par des entiers (ou réels). Pour qu'il existe un flot entier (ou réel) respectant les contraintes de capacité :

$$a \leq \varphi \leq b$$

il faut et il suffit que l'une des conditions suivantes soit satisfaite :

a) quelle que soit la tension θ il existe un vecteur f_θ orthogonal à θ et compatible avec les limites (a, b)

$$< f_\theta, \theta > = 0$$

$$a \leq f_\theta \leq b.$$

b) Quel que soit le cocycle élémentaire U_A :

$$\sum_{i \in U_{A^+}} a_i - \sum_{i \in U_{A^-}} b_i \leq 0$$

et

$$\sum_{i \in U_{A^-}} a_i - \sum_{i \in U_{A^+}} \leq 0.$$

Le théorème est une généralisation du théorème de Hoffmann [2] .

De la 2ème condition du théorème précédent, on peut dire que la conservation du flot exige que le flux minimum entrant dans tout ensemble

de sommets A soit inférieur au flux sortant et il suffit de vérifier cette condition à travers les seules coupes du graphe pour l'existence d'un tel flot.

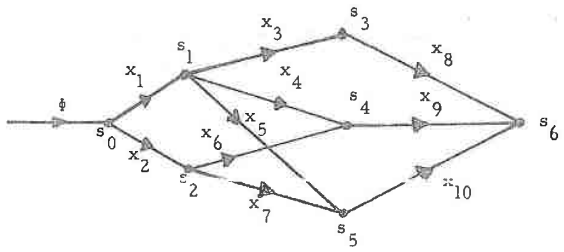
Alors, on choisit les arcs qui constituent un chemin ou plusieurs qui relient l'entrée du graphe avec sa sortie et vérifient la condition précédente. Ces arcs représentent les variables de base de début.

Dans notre problème, la contrainte (cf. I.5.3) :

$$\phi - \sum_{u \in U_{x_0^+}} \varphi(u) = 0$$

nous oblige à choisir au moins un arc (une variable) parmi $U_{s_0^+}$, l'ensemble des arcs qui sortent de s_0, l'entrée du graphe. Le reste s_0^+ des variables sera choisi entre les autres contraintes bilatérales du problème à condition que chaque contrainte soit représentée au moins par une variable ; et que les variables (arcs) choisies forment entre-elles, un chemin reliant s_0 avec s_n la sortie du graphe.

Par exemple, considérons que le graphe suivant respecte les conditions du théorème précédent :



On peut constater que chaque ensemble d'arcs qui représente les variables de base X_B et relie l'entrée du graphe avec sa sortie rend la

matrice $\mathcal{C}'_B$ inversible. Une solution peut être trouvée en choisissant X_B tel que :

$x_1, x_3, x_8, x_2, x_6, x_{10}$;

$x_1, x_5, x_8, x_2, x_6, x_9$;

$x_1, x_4, x_9, x_2, x_3, x_7$;

$x_1, x_2, x_8, x_9, x_{10}, x_4$;

$x_1, x_2, x_8, x_9, x_{10}, x_5$;

$x_1, x_2, x_8, x_9, x_{10}, x_7$.

Ce ne sera pas le cas si les arcs choisis ne contiennent aucun arc de $U_{s_6^-}$. Par exemple :

$x_1, x_2, x_4, x_5, x_6, x_7$;

$x_2, x_3, x_4, x_5, x_6, x_7$.

PARTIE II

BIBLIOTHEQUE DES SOUS-PROGRAMMES

ET NOTICES

MISE EN ŒUVRE SUR QUELQUES EXEMPLES.

II - 1 DESCRIPTION DU PROGRAMME PRINCIPAL ET DES SOUS-PROGRAMMES UTILISES :

La mise en œuvre de la technique du gradient projeté demande l'utilisation de dix sous-programmes dont quatre sous-programmes caractérisent le problème à traiter. Ces quatre sous-programmes doivent être réécrits pour chaque problème. Le programme principal et ces sous-programmes sont écrits en FORTRAN IV de base. Ils ont été testés sur un ordinateur MITRA 125. Malgré les dimensions réduites de la mémoire (32 K mots), nous avons pu traiter des problèmes importants en utilisant les techniques du recouvrement.

II.1.a Sous-programmes de description du problème :

Les quatre sous-programmes sont :

- FUNCTION F (X) : qui calcule la fonction objectif
- GRADS : qui calcule le gradient simple de la fonction objectif
- CONT : selon l'indicateur K, il calcule :
  - les valeurs prises en X par les contraintes non saturées, si K = 1 ;
  - les valeurs prises en X par les contraintes "utiles", si K = 0.

Les valeurs des contraintes calculées seront rangées en séquence dans un tableau C. Il calcule aussi pour chaque contrainte la somme des gradients simples de la fonction objectif par rapport aux variables de cette contrainte. La valeur sera rangée dans le tableau DD. Par exemple, soit la contrainte I sous la forme :

$$C(I) = X_1 + X_3 + X_4 - X_7 \text{ alors}$$

$$DD(I) = \partial f/\partial x_1 + \partial f/\partial x_3 + \partial f/\partial x_4 - \partial f/\partial x_7.$$

Nous aurons besoin de faire ce calcul, pour le calcul de la valeur de h qui sature une contrainte (cf. I-7).

- DEPCØ : Il range par colonne les dérivées d'une même contrainte utile. La matrice G contiendra ${}^t[\partial C_i/\partial x_s]$

II.1.b Autres sous-programmes

Les six autres sous-programmes sont :

- GPRØJ, MCADIR } (cf. I-4)
- ELNUML, RESYS } (cf. I-6)
- ETAPL (cf. I-7)
- CRBAS 1 ou 2 (cf. I-8).

Rappelons que le sous-programme ETAPL comprend deux sous-programmes :

a) le sous-programme modifié SATUR de la bibliothèque, après le remplacement de la technique de dichotomie par la méthode décrite en (I-7), qui correspond au type des contraintes linéaire ;

b) le sous-programme de bibliothèque ETAP2.

Des instructions "CØMMØN" étiquetées contenant les paramètres, les tableaux de travail et les identificateurs globaux du problème, seront ajoutées pour relier tous les sous-programmes entre eux et ceux du programme "appelant". Nous indiquerons d'abord le rôle des paramètres utilisés et donnerons ensuite les listes d'instructions des divers sous-programmes.

II - 2 LISTE DES VARIABLES ET TABLEAUX UTILISES

A - Les Variables

N : nombre de variables.

NT : nombre total de contraintes.

M : nombre de variables de base (ou également le nombre de contraintes utiles).

INCI : indicateur d'incident.

IFI : indicateur de fin d'itération. Il prend la valeur 0 lorsque le maximum est atteint.

ICL : indicateur de contrainte libérée et a la valeur 1 si une contrainte a été libérée et 0 dans le cas contraire.

NA : paramètre de surdimentionnement (= Max (N, NT)). Indispensable pour MCADIR.

P : nombre de contraintes bilatérales ; il doit être déclaré "entier".

NCVB : nombre de changements de la liste de variables de base (égale à 0 au début du programme).

NREP : nombre de cases de l'anneau (cf. I-8).

FX : contient en permanence la valeur $f(X^r)$ du premier itéré obtenu.

QE : le flux entrant dans le graphe.

H et H1 : Pas de montée. H est fixe ; H1 évolue par le programme.

NS : nombre de sources (pour le problème de transport).

ND : nombre de demandes ou points de consommation (pour le problème de transport).

B - Les Tableaux

D (N) : contient la direction de montée.

ILU (NT) : ILU (I) = 1 si la contrainte C_i est bilatérale ou unilatérale saturée.
ILU (K) = 0 si la contrainte C_k est unilatérale vérifiée non saturée.

IVL (N) : IVL (H) = 0 si x_h est variable hors-base

IVL (B) = 1 si x_b est variable de base.

IVX (NT) : IVX (I) = J si la contrainte C_i dépend de la seule variable x_j.

IVX (I) = 0 dans les autres cas.

EPS (2) : ε_1, ε_2 utilisées dans les tests des sous-programmes CPRØJ et ETAPL.

G (NT, NT) : contient les dérivées $\partial C_i / \partial x_s$ des contraintes utiles par rapport à toutes les variables.

X (N) : contient en permanence le dernier itéré X^r. Il doit être initialisé par le programme principal.

C1 (N) et C2 (N) : Deux coefficients utilisés dans la fonction objectif représentant le gain ou la perte (selon la nature du problème Max . F ou Min . F) pour chaque arc du graphe.

IS (NREP) : le contenu des cases de l'anneau (cf. I-8).

NVB (M) : nombre de variables dans chaque contrainte bilatérale (à utiliser avec CRBAS2 (cf. I-8)).

LT (M, K) : tableau pour ranger les variables qui se trouvent dans les contraintes bilatérales (cf. I-8).

C (NT)
Y (N), Z (N), U (N)
FP (N)
IP (NA)
W (NT, NT)

} Tableaux de travail.

U1 (NA), U2 (NA), U3 (NA) : tableaux de travail (utilisés en MCADIR).

DD (NT) : tableau de travail utilisé en CØNT.

C - Les instructions CØMMØN

Nous indiquons ci-après les identificateurs de chaque instruction

CØMMØN, et donnons la liste des CØMMØN utilisés dans chaque sous-programme.

CØMMØN |A| N, NT, M, INCI, IFI, ICL, D(N), ILU(NT), IVL(N), IVX(NT), EPS(2), C(NT), G(N, N), U1(NA), FP(NT), W(N, N), U2(NA), U3(NA), IP(NA), NA.

CØMMØN |A1| X (N)

CØMMØN |B| FX, H, H1, Y(N), Z(N), U(N)

CØMMØN |D1|NVB(P), LT(P, K), P, IS(P)

CØMMØN |E |FY, FZ, FU, DD(NT), QE

CØMMØN |E0| FY, FZ, FU, DD(NT)

CØMMØN |E1|C1(N), C2(N), IS(NREP)

CØMMØN |E2|C1(N), C2(N), NS, ND

Problème de / Sous-Programme	Transport	flot
CONT	A, E0	A, E
CRBAS1	A	A
CRBAS2	A	A
DEPCØ	—	—
ELNUML	A	A
ETAPL	A, A1, B, E0	A, A1, B, E
FUNCTION F(X)	E2	E1
GPRØJ	A, A1	A, A1
GRADS	E2	E1
MCADIR	—	—
RESYS	—	—

II - 3 DESCRIPTION DES SOUS-PROGRAMMES DE BIBLIOTHEQUE

Dans cette partie, nous décrivons les sous-programmes de bibliothèque qui mettent en œuvre les techniques décrites en (I.6.3), (I-7) et (I-8). Les sous-programmes sont :

- RESYS
- CRBAS1
- CRBAS2
- ETAPL

et - ELNUML.

Les deux sous-programmes (GPRØJ et MCADIR) sont ceux de la bibliothèque [16] .

Le sous-programme RESYS :

Ce sous-programme résoud un système W * X = C de M équations linéaires à M inconnues lorsque la matrice W est unimodulaire.

Paramètres utilisés :

M : Nombre des variables de base.

W : Matrice carrée d'ordre M. Les coefficients de la matrice sont les dérivées partielles des contraintes utiles par rapport aux variables de base.

C : Vecteur d'ordre M qui contient les valeurs prises en X par les contraintes utiles. En fin d'exécution et en cas de déroulement normal, la solution du système sera rangée dans C.

LP : Indicateur d'incident :

LP = 0 en cas de déroulement normal

LP = 1 indique que la matrice W est non inversible.

Sous-programmes appelés : aucun

```
      SUBROUTINE RESYS(W,C,M,LP)
      DIMENSION W(30,1),C(1)
      NP=M+1
      DO 11 I=1,M
   11 W(I,NP)=C(I)
      DO 23 K=1,M
      IF(W(K,K).NE.0.) GO TO 16
C
C     LE PIVOT EST EGAL A ZERO.
C
      NN=K
   13 NN=NN+1
      IF(NN.LE.M) GO TO 14
      LP=1
      RETURN
   14 IF(W(NN,K).EQ.0.) GO TO 13
C
C     LE REMPLACEMENT DE LA LIGNE K PAR LA LIGNE NN.
C
      DO 15 L=K,NP
      R=W(K,L)
      W(K,L)=W(NN,L)
   15 W(NN,L)=R
   16 IF(W(K,K).GT.0) GO TO 18
      DO 17 I=K,NP
   17 W(K,I)=-W(K,I)
   18 DO 23 K1=1,M
      IF(K1.EQ.K) GO TO 23
      IF(W(K1,K))19,23,21
C
C     L'ADDITION DE LA LIGNE DU PIVOT A LA LIGNE K1.
C
   19 DO 20 L=1,NP
   20 W(K1,L)=W(K1,L)+W(K,L)
      GO TO 23
C
C     LA SOUSTRACTION DE LA LIGNE K1 A LA LIGNE DU PIVOT.
C
   21 DO 22 L=1,NP
   22 W(K1,L)=W(K1,L)-W(K,L)
   23 CONTINUE
      DO 24 I=1,M
   24 C(I)=W(I,NP)
      LP=0
      RETURN
      END
```

Le sous-programme CRBAS1 :

Ce sous-programme change la liste des variables de base automatiquement en appliquant la première méthode décrite en (I-8).

Paramètres utilisés :

ITT : Numéro de la case de l'anneau d'où on commence la recherche des variables de base.

P : Entier, représente le nombre des contraintes bilatérales. Sa valeur doit être fixée dans le programme principal.

NREP : Nombre des cases de l'anneau, c'est-à-dire le nombre des variables présentes dans les contraintes bilatérales. On compte R cases si une variable est répétée R fois dans ces contraintes.

IS : Vecteur de dimension NREP qui contient les variables qui se trouvent dans le système des contraintes bilatérales, rangées comme indiqué en (I-8).

Sous-programmes appelés :

aucun.


```
      SUBROUTINE CRBAS1(ITT,P,NREP,IS)
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(10),ILU(30),IVL(10),IVX(30),EPS(2),
     1C(30),G(30,30),U1(30),FP(10),W(30,30),U2(30),U3(30),IP(30),NA
      DIMENSION IS(1)
      INTEGER P
      K=0
C
C     REMISE A ZERO LE TABLEAU QUI CONTIENDRA LES VARIABLES DE BASE.
C
      DO 1 I=1,N
    1 IVL(I)=0
C
C     LA RECHERCHE DES CONTRAINTES BILATERALES ET UNILATERALES SATUREES
C     QUI DEPEND D'UNE SEULE VARIABLE ET LA METTRE DANS LA BASE.
C
      DO 100 I=1,NT
      IF(ILU(I).EQ.0) GO TO 100
      IF(IVX(I).EQ.0) GO TO 100
      K=K+1
      JJ=IVX(I)
      IVL(JJ)=1
  100 CONTINUE
      L=ITT-1
C
C     LA RECHERCHE A PARTIR DE LA CASE ITT DE L'ANNEAU.
C
      DO 20 I=1,NREP
      L=L+1
      IF(L.LE.NREP) GO TO 10
C
C     ON RECOMMENCE LA RECHERCHE A PARTIR DE LA 1 ERE CASE DE L'ANNEAU.
C
      L=1
   10 K1=IS(L)
      IF(IVL(K1).EQ.0) GO TO 15
      GO TO 20
   15 IVL(K1)=1
      K=K+1
      IF(K.EQ.M) GO TO 30
C
C     LE NOMBRE DES VARIABLES DE BASE INFERIEUR AU NOMBRE DES CONTRAINTES
C     UTILES.
C
   20 CONTINUE
   30 RETURN
      END
```

Le sous-programme CRBAS2 :

Ce sous-programme change les variables de base en appliquant la deuxième méthode décrite en (I-8).

Paramètres utilisés :

NVB :
LT :
IS :
P :
} (cf. II-2)

Sous-programmes appelés :

aucun.

```
      SUBROUTINE CRBAS2(NVB,LT,IS,P)
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(30),ILU(42),IVL(30),IVX(42),EPS(2),
     1C(42),G(30,30),U1(30),FP(42),W(30,30),U2(30),U3(30),IP(30),NA
      DIMENSION NVB(1),LT(12,1),IS(1)
      INTEGER P
      ISW=0
      INCI=0
      DO 1 I=1,N
    1 IVL(I)=0
      DO 100 I=1,NT
      IF(ILU(I).EQ.0) GO TO 100
      IF(IVX(I).EQ.0) GO TO 100
      JJ=IVX(I)
      IVL(JJ)=1
  100 CONTINUE
      K=0
    5 DO 20 I=1,P
   10 ISS=IS(I)
      K1=LT(I,ISS)
      IF(IVL(K1).EQ.0) GO TO 15
      ISS=ISS+1
      IF(ISS.GT.NVB(I)) GO TO 20
      IS(I)=IS(I)+1
      GO TO 10
   15 IVL(K1)=1
      K=K+1
      IF(K.EQ.P) GO TO 35
   20 CONTINUE
      IF(ISW.EQ.1) GO TO 30
      DO 25 I=1,P
   25 IS(I)=I
      ISW=1
      GO TO 5
   30 INCI=21
   35 RETURN
      END
```


Le sous-programme ETAPL :

Le sous-programme suppose une direction de montée D connue et calcule un vecteur X^{r+1} tel que $f(X^{r+1}) > f(X^r)$ et le pas h_1 correspondant. Dans le cas où X^{r+1} ne vérifie pas toutes les contraintes unilatérales non saturées, le vecteur X^{r+1} sera obtenu par la saturation d'une nouvelle contrainte en calculant le pas h_1 comme expliqué en (I-7).

Ce sous-programme regroupe les deux sous-programmes de la bibliothèque :

- SATUR : après la modification qui consiste à remplacer la méthode de dichotomie par la méthode décrite en (I-7) ;
- ETAP2 : sous-programme de la bibliothèque sans modification.

Paramètres utilisés :

aucun.

Sous-programmes appelés :

ELNUML - CØNT - F.

Indicateur d'incident INCI qui vaut :

0 : en cas de déroulement normal ;

51 : si $h_1 \leq \varepsilon_2$;

52 : en cas d'incident dans le calcul de Y par ELNUML ;

61 : dans le cas où aucune contrainte n'a été saturée, le pas calculé étant égal à ∞ ;

62 : en cas d'incident dans le calcul de Y par ELNUML, où Y a été calculé par le pas h_1 trouvé par la méthode décrite en (I-7).

```
      SUBROUTINE ETAPL
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(30),ILU(42),IVL(30),IVX(42),EPS(2),
     1C(42),G(30,30),U1(30),FP(42),W(30,30),U2(30),U3(30),IP(30),NA
      COMMON/A1/X(30)
      COMMON/B/FX,H,H1,Y(30),Z(30),U(30)
      COMMON/D1/NVB(12),LT(12,10),P,IS(12)
      COMMON/E0/FY,FZ,FU,DD(42)
      INTEGER P
      H1=H
    1 DO 2 I=1,N
    2 Y(I)=X(I)+H1*D(I)
      CALL ELNUML(Y)
      IF(INCI.GE.0) GO TO 101
```

```
      N1=NT-M
      IF(N1.EQ.0) GO TO 19
      CALL CONT(Y,1)
      DO 5 K=1,N1
      IF(C(K).LT.0.) GO TO 20
    5 CONTINUE
   19 FY=F(Y)
      IF(FY.GT.FX) GO TO 50
      H1=H1*0.25
      IF(H1.LE.EPS(2)) GO TO 100
      GO TO 1
C
C     CALCUL DU PAS H1.
C
   20 DO 13 I=1,N1
   13 PP(I)=C(I)
      CALL CONT(X,1)
      ISC=0
C
C     REMISE A ZERO LE NOMBRE DES CONTRAINTES NON VERIFIES.
C
   25 NCNV=0
C     LA RECHERCHE DE MIN. H
      SS=999.E+09
      L=1
      DO 110 I=1,N1
      IF(ILU(I).EQ.1) GO TO 110
      IF(PP(L).GE.0.) GO TO 109
      NCNV=NCNV+1
      S=-C(L)/DD(L)
      IF(S.LE.0.) GO TO 110
      IF(SS.LT.S) GO TO 110
      SS=S
      KK=L
      ISC=1
  109 L=L+1
  110 CONTINUE
      IF(ISC.EQ.0) GO TO 1010
      HH=SS
      IF(NCNV.EQ.1) GO TO 44
      IF(HH.GT.EPS(2)) GO TO 44
      C(L)=0.0
      GO TO 25
   44 DO 14 I=1,N
   14 Y(I)=X(I)+HH*D(I)
      CALL ELNUML(Y)
      IF(INCI.NE.0) GO TO 1000
      CALL CONT(Y,1)
      K=0
      DO 16 K=1,N1
      IF(ABS(C(K)).GE.0.00001) GO TO 16
      KK=K
      KC=KC+1
   16 CONTINUE
      IF(KC.LE.1)GO TO 120
      GO TO 20
C
C     SATURATION D'UNE CONTRAINTE.
C
  120 FY=F(Y)
      IF(FY.LE.FX) GO TO 1010
```



```
      K=0
      DO 60 I=1,NT
      IF(ILU(I).EQ.1) GO TO 60
      K1=I
      K=K+1
      IF(K.EQ.KK) GO TO 200
   60 CONTINUE
  200 ILU(K1)=1
C
C     RECHERCHE D'UNE VARIABLE ENTRANTE.
C
      M=M+1
      J1=IVX(K1)
      IF(J1.LE.0) GO TO 210
      IF(IVL(J1).NE.0) GO TO 210
      IVL(J1)=1
      GO TO 300
  210 DO 70 I=1,N
      IF(IVL(I).EQ.1) GO TO 70
      IVL(I)=1
      GO TO 300
   70 CONTINUE
  300 DO 80 I=1,N
   80 X(I)=Y(I)
      H1=HH
      FX=F(X)
      RETURN
 1000 INCI=62
      RETURN
 1010 INCI=61
      RETURN
   50 H2=2.*H1
      DO 5 I=1,N
    5 Z(I)=X(I)+H2*D(I)
      CALL ELNUML(Z)
      IF(INCI.EQ.0) GO TO 511
   15 FX=FY
      DO 6 I=1,N
    6 X(I)=Y(I)
      INCI=0
      RETURN
  511 IF(M1.EQ.0) GO TO 22
      CALL CONT(Z,1)
      DO 7 K=1,N1
      IF(C(K).LT.0.) GO TO 15
    7 CONTINUE
   22 FZ=F(Z)
      DEL=FX-2.*FY+FZ
      IF(DEL.LT.0.) GO TO 55
   56 IF(FY.GT.FZ) GO TO 15
      FX=FZ
      H1=H2
      INCI=0
      DO 8 I=1,N
    8 X(I)=Z(I)
      RETURN
```

```
 55 HE=H1*(3.*FX-4.*FY+FZ)*0.5/DEL
    DO 9 I=1,N
  9 U(I)=X(I)+HE*D(I)
    CALL ELNUML(U)
    IF(INCI.NE.0) GO TO 56
    IF(H1.EQ.0) GO TO 23
    CALL CONT(U,1)
    DO 10 K=1,M1
    IF(C(K).LT.0.) GO TO 56
 10 CONTINUE
 23 FU=F(U)
    IF(FU.LT.FZ) GO TO 56
    IF(FU.LT.FY) GO TO 56
    FX=FU
    H1=HE
    INCI=0
    DO 11 I=1,N
 11 X(I)=U(I)
    RETURN
100 INCI=51
    RETURN
101 INCI=52
    RETURN
    END
```


Le sous-programme ELNUML :

Ce sous-programme calcule les valeurs des variables de base X_B^{r+1} à partir des variables indépendantes X_H^{r+1} pour le système des contraintes bilatérales et unilatérales saturées

$$\mathcal{C}(X_H^{r+1}, X_B^{r+1}) = 0 \qquad (34)$$

La technique du calcul est décrite en (I-6).

Paramètres utilisés :

X : Un tableau d'ordre N qui contient les variables de base et celles hors base. Aprés l'exécution et en cas de déroulement normal, le tableau X contient la solution du système ci-dessus.

Sous-programmes appelés :

CØNT, DEPCØ, RESYS.

```
    SUBROUTINE ELNUML(X)
    COMMON/A/N,NT,M,INCI,IFI,ICL,D(30),ILU(42),IVL(30),IVX(42),EPS(2),
   1C(42),G(30,30),U1(30),FP(42),W(30,30),U2(30),U3(30),IP(30),NA
    DIMENSION X(1)
    IF(M.EQ.0) GO TO 5
    CALL CONT(X,0)..... Le calcul des valeurs des contraintes utiles.
    CALL DEPCO(G,ILU)..... Le calcul de C' pour toutes les contraintes.
    K=0
    DO 2 I=1,N
    IF(IVL(I).EQ.0) GO TO 2
    K=K+1
    DO 3 J=1,M
  3 W(J,K)=G(I,J)..... Le rangement de C'B en W.
  2 CONTINUE
    CALL RESYS(W,C,M,LP)
    IF(LP.NE.0) GO TO 100
    K=0
    DO 4 I=1,N
    IF(IVL(I).EQ.0) GO TO 4
    K=K+1
    X(I)=X(I)-C(K)..... Le calcul de X_B^{r+1}.
  4 CONTINUE
  5 INCI=0
    RETURN
100 INCI=11
    RETURN
    END
```

II - 4 APPLICATION DETAILLEE SUR TROIS EXEMPLES :

Trois exemples sont choisis à titre d'application. Le premier exemple traite un problème de flot de 10 variables (arcs). Le deuxième exemple représente un problème de transport de trois sources et dix points de consommation déjà traité en utilisant la technique de la programmation dynamique [1] p. 92. Le troisième exemple traite le cas général du problème de transport, dans lequel :

$$\sum_{i=1}^{N} P_i \neq \sum_{j=1}^{M} R_j .$$

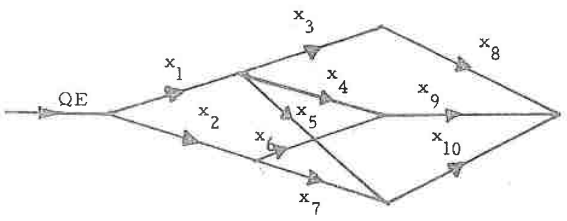
Nous nous limiterons dans ce paragraphe à l'étude des programmes de description (cf. II.1.a), et des programmes généraux pour les deux premiers exemples. Les résultats des trois exemples apparaitront en (II.6.1, II.6.2, II.6.4).

II.4.1 Enoncé du problème de flot :

Etant donné un flux fixe QE entrant dans un graphe orienté (fig. 9) dans lequel :

1 - chacun de ces arcs est limité par une capacité donnée ;

2 - pour chaque arc, il existe un coût.



(fig. 9)

Le problème est de trouver le flux x_i qui traverse chaque arc u_i, de façon à minimiser la fonction objectif :

$$F = \sum_{j=1}^{10} (C_j^1 x_j + C_j^2 x_j^2).$$

Bien entendu la somme des flux sortant du graphe est égale à QE. Les autres contraintes de ce problème sont décrites en (I.5.3).

Nous avons pris les dix premières valeurs de C^1 et C^2 du problème de transport pour les utiliser comme données dans le problème actuel (cf. II.4.3) représenté par le graphe de la figure (9).

II.4.2 Phase d'initialisation du problème dans le programme principal

Les données du problème seront :

N = 10, NT = 26, M = P = 6, ε_1 = 0.01, ε_2 = 1.E-10, NA = 30, NREP = 17.

Les tableaux ILU, IVX, IVL seront initialisés comme suit :

ILU : $\overbrace{1 \quad 1 \ldots 1}^{\text{M fois}} \quad \overbrace{0 \quad 0 \quad 0 \ldots . \quad 0}^{\text{(NT-M) fois}}$.

Le nombre des contraintes unilatérales est le double du nombre des arcs (voir relation (17')).

IVX : $\overbrace{0 \quad 0 \ldots 0}^{\text{M fois}} \quad 1 \quad 2 \quad 3 \ldots \quad 10 \quad 1 \quad 2 \quad 3 \ldots 10$

IVL : Le sous-programme CRBAS (1 ou 2) choisira M variables et les fera entrer dans la base en posant IVL de chaque variable choisie égal à 1, et 0 ailleurs.

II. 4. 3 Programmes de description du problème de flot :

```
      FUNCTION F(X)
      COMMON/E1/C1(30),C2(30),IS(17)
      DIMENSION X(1)
      F=0.
      DO 10 J=1,10
   10 F=F-C1(J)*X(J)-C2(J)*X(J)*X(J)
      RETURN
      END
```

```
      SUBROUTINE GRADS(X,FP)
      COMMON/E1/C1(30),C2(30),IS(17)
      DIMENSION X(1),FP(1)
      DO 1 J=1,10
    1 FP(J)=-C1(J)-2.*C2(J)*X(J)
      RETURN
      END
```

En considérant le graphe de la figure (9), les contraintes bilatérales seront :

$$x_1 - x_3 - x_4 - x_5 = 0$$

$$x_2 - x_6 - x_7 = 0$$

$$x_3 - x_8 = 0$$

$$x_4 - x_9 = 0$$

$$x_5 + x_7 - x_{10} = 0$$

$$x_1 + x_2 - QE = 0.$$

Pour écrire les contraintes unilatérales représentées par (17'), il sera nécessaire de connaître la capacité p (u) de chaque arc u du graphe. On considère que les capacités des arcs du graphe précédent seront successivement :

$$p(u_i) = 35 - 30 - 15 - 10 - 15 - 20 - 15 - 15 - 25 - 25.$$

$i = 1, \ldots, 10.$

Les contraintes unilatérales seront :

$$\left.\begin{array}{l} -x_i + p(u_i) \geq 0 \\ x_i \geq 0 \end{array}\right\} \quad i = 1, 2, \ldots, 10$$

et

```
      SUBROUTINE CONT(X,K)
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(10),ILU(30),IVL(10),IVX(30),EPS(2),
     1C(30),G(30,30),U1(30),FP(10),W(30,30),U2(30),U3(30),IP(30),NA
      COMMON/E/FY,FZ,FU,DD(30),QE
      DIMENSION X(1)
      C(1)=X(1)-X(3)-X(4)-X(5)
      DD(1)=D(1)-D(3)-D(4)-D(5)
      C(2)=X(2)-X(6)-X(7)
      DD(2)=D(2)-D(6)-D(7)
      C(3)=X(3)-X(8)
      DD(3)=D(3)-D(8)
      C(4)=X(4)+X(6)-X(9)
      DD(4)=D(4)+D(6)-D(9)
      C(5)=X(5)+X(7)-X(10)
      DD(5)=D(5)+D(7)-D(10)
      C(6)=X(1)+X(2)-QE
      DD(6)=D(1)+D(2)
      C(7)=-X(1)+35.
      C(8)=-X(2)+30.
      C(9)=-X(3)+15.
      C(10)=-X(4)+10.
      C(11)=-X(5)+15.
      C(12)=-X(6)+20.
      C(13)=-X(7)+15.
      C(14)=-X(8)+15.
      C(15)=-X(9)+25.
      C(16)=-X(10)+25.
      J=7
      DO 1 I=1,10
      L=I+16
      DD(J)=-D(I)
      C(L)=X(I)
      DD(L)=D(I)
      J=J+1
    1 CONTINUE
      L=0
      DO 3 I=1,NT
      IF(ILU(I).EQ.K) GO TO 3
      L=L+1
      C(L)=C(I)
      DD(L)=DD(I)
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DEPCO(G,ILU)
      DIMENSION G(30,1),ILU(1)
      DO 100 I=1,30
      DO 100 J=1,30
  100 G(I,J)=0.
      I=1
      IF(ILU(1).EQ.0) GO TO 1
      G(1,I)=1.
      G(3,I)=-1.
      G(4,I)=-1.
      G(5,I)=-1.
      I=I+1
    1 IF(ILU(2).EQ.0) GO TO 2
      G(2,I)=1.
      G(6,I)=-1.
      G(7,I)=-1.
      I=I+1
    2 IF(ILU(3).EQ.0) GO TO 3
      G(3,I)=1.
      G(8,I)=-1.
      I=I+1
    3 IF(ILU(4).EQ.0) GO TO 4
      G(4,I)=1.
      G(6,I)=1.
      G(9,I)=-1.
      I=I+1
    4 IF(ILU(5).EQ.0) GO TO 5
      G(5,I)=1.
      G(7,I)=1.
      G(10,I)=-1.
      I=I+1
    5 IF(ILU(6).EQ.0) GO TO 6
      G(1,I)=1.
      G(2,I)=1.
      I=I+1
    6 K=7
      DO 8 L=1,10
      IF(ILU(K).EQ.0) GO TO 7
      G(L,I)=-1.
      I=I+1
    7 K=K+1
    8 CONTINUE
      K=17
      DO 10 L=1,10
      IF(ILU(K).EQ.0) GO TO 9
      G(L,I)=1.
      I=I+1
    9 K=K+1
   10 CONTINUE
      RETURN
      END
```


II.4.4 Programme principal dans le cas du problème de flot :

On utilisera dans ce programme la première méthode de changement de base CRBAS1. Les données et les paramètres sont les mêmes qu'en (II.4.1) et (II.4.2).

Le programme principal sera écrit comme suit :

```
      COMMON/A/N,NT,M,INC1,IFI,ICL,D(10),ILU(30),IVL(10),IVX(30),EPS(2),
     1C(30),G(30,30),U1(30),FP(10),W(30,30),U2(30),U3(30),IP(30),NA
      COMMON/A1/X(10)
      COMMON/B/FX,H,H1,Y(10),Z(10),U(10)
      COMMON/E/FY,FZ,FU,DD(30),QE
      COMMON/E1/C1(30),C2(30),IS(17)
    3 FORMAT(2513)
   13 FORMAT(' IVL=',10I3,9X,I3,5X,I3)
 1000 FORMAT(' ICT=',I3/' X=',10F10.6/' D=',10F10.6//'  FX=',F15.7//' ILU
     1=',26I2/' IVL=',10I2/)
 2000 FORMAT(16F5.2)
 3000 FORMAT('    MAX. OBTENU ',/' X OPT.=',10F8.4//'  F OPT.=',F20.8//'
     1 NO. DES ITERATIONS =',I4,5X,'NCVB=',I3/' D=',10F10.6)
 5000 FORMAT('***** PLUS DE CHANGEMENT DE VAR. DE BASE')
C
C     PARAMETRES ET DONNEES DU PROBLEME.
C
      INTEGER P
      NA=30
      NT=26
      N=10
      NREP=17
      P=6
      M=P
      NCVB=0
      FY=0.
      FZ=0.
      FU=0.
      EPS(1)=0.01
      EPS(2)=1.E-10
      DO 77 I=1,M
   77 IVX(I)=0
      M1=N+1
      M2=N+N
      DO 88 I=M1,M2
      J=I-M
   88 IVX(I)=J
      M3=M2+1
      DO 99 I=M3,NT
      J=I-M2
   99 IVX(I)=J
```

```
      DO 4 I=1,M
    4 ILU(I)=1
      DO 5 I=M1,NT
    5 ILU(I)=0
      ITT=0
      ICT=0
       M1=M
      READ(105,2000)M,M1
      WRITE(108,2000)M,M1
C
C     LECTURE DES COEFFICIENTS C1 ET C2 DE LA FONCTION ECONOMIQUE.
C
      READ(105,2000)(C1(J),J=1,N)
      READ(105,2000)(C2(J),J=1,N)
      WRITE(108,2000)(C1(J),J=1,N)
      WRITE(108,2000)(C2(J),J=1,N)
C
C     LECTURE DU FLOT ENTRANT DANS LE GRAPHE.
C
      READ(105,2000)QE
C
C     LECTURE D'UN VECTEUR X COMPATIBLE AVEC LES CONTRAINTES.
C
      READ(105,2000)(X(J),J=1,N)
      WRITE(108,2000)(X(J),J=1,N)
      DO 6 I=1,N
      IF(X(I).GT.0.0) GO TO 6
      M=M+1
      M2=M1+I
      ILU(M2)=1
    6 CONTINUE
C
C     LECTURE DU CONTENU DE L'ANNEAU.
C
      READ(105,3)(IS(I),I=1,NREP)
      WRITE(108,3)(IS(I),I=1,NREP)
C
C     L'APPLICATION DE LA 1'ERE METHODE DU CHANGEMENT DES VARIABLES DE
C     BASE.
C
   11 ITT=ITT+1
      IF(ITT.LE.NREP) GO TO 7
C
C     L'ARRIVEE A LA DERNIERE CASE DE L'ANNEAU.
C
C     TEST SI LE NOMBRE DES CONTRAINTES UTILES M NE CHANGE PAS.
C
      IF(M1.EQ.M) GO TO 190
      M1=M
      ITT=ITT-NREP
C
C     ON RECOMMENCE A LA 1 ERE CASE DE L'ANNEAU.
C
    7 CALL CRBAS1(ITT,P,NREP,IS)
```



```
C
C     LE NOMBRE DE CHANGEMENT DES VARIABLES DE BASE AUGMENTE D'UN.
C
      NCVB=NCVB+1
    1 CALL GPROJ(P)
      IF(INCI.NE.0) GO TO 100
      FX=F(X)
      IF(IFI.EQ.0) GO TO 10
      CALL ETAPL
      IF(INCI.NE.0) GO TO 100
      ICT=ICT+1
      WRITE(108,1000)ICT,(X(I),I=1,N),(D(I),I=1,N),FX,(ILU(I),I=1,NT),(I
     1VL(I),I=1,N)
      GO TO 1
   10 FX=F(X)
      WRITE(108,3000)(X(I),I=1,N),FX,ICT,NCVB,(D(I),I=1,N)
      GO TO 19
  100 WRITE(108,13)(IVL(I),I=1,N),INCI,NCVB
      GO TO 11
  190 WRITE(108,5000)
   19 STOP
      END
```

II - 5 ENONCE DU PROBLEME DE TRANSPORT

Il s'agit de trouver les inconnues X_i, i = 1, 2, ..., 30 qui minimisent la fonction :

$$F = \sum_{i=1}^{30} (C_i^1 x_i + C_i^2 x_i^2)$$

sous les contraintes représentées par (15) en (I.5.1.a) où :

$$q_1 = 100, \quad q_2 = 97, \quad q_3 = 138 \text{ et}$$

Q_i (i = 1, 2, ..., 10) ont les valeurs : 25, 40, 60, 30, 20, 30, 35, 30, 25, 40,.

Les coefficients C^1 et C^2 sont :

C^1	1	2	3	1.5	2.5	5	3	6	6	6	3.1	4.1	2.1	1.1	2.6	3	1	2	2	5	7	3	9	1	1	2	4	3	5	6
C^2	.2	.06	.01	0	.05	.01	0	0	.05	0	.1	0	0	.1	0	0	.2	0	0	.01	0	.04	0	0	.2	0	0	.1	0	0

La fonction objectif devient :

$$\text{Max} - F = - \sum_{i=1}^{30} (C_i^1 x_i + C_i^2 x_i^2).$$

Le vecteur initial X est :

8	20	35	5	0	5	10	5	7	5	100
10	5	15	15	10	17	5	0	10	10	97
7	15	10	10	10	8	20	25	8	25	138
25	40	60	30	20	30	35	30	25	40	

II.5.1 Phase "d'initialisation" du problème dans le programme principal

Les données du problème seront :

N = 30, NT = 42, M = P = 12, NS = 3, ND = 10, $\varepsilon_1 = 0.005$, $\varepsilon_2 = 0.01$,
NA = 42, (NVB (J), J = 1, 12) = (10, 10, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3).
Le tableau LT (I, J) sera donné comme suit :

```
 1   2   3 ..... 10
11  12  13 ..... 20
 1  11  21
 2  12  22
 .
 .
 .
10  20  30
```

Les tableaux ILU, IVX, et IVL seront initialisés selon le type de contraintes (bilatérales et unilatérales saturées ou unilatérales non saturées) et le type de variables (celles qui sont choisies pour entrer dans la base et le reste des variables). Dans notre problème les tableaux seront initialisés comme suit :

ILU : 1 1....1 (M fois) 0 0....0 ((NT-M) fois) (les M premières contraintes sont de type bilatérales)

IVX : 0 0....0 (30 fois) 1 2...30 (les contraintes de signe qui ne dépendent que d'une seule variable)

IVL : Comme dans le cas du problème de flot.

Enfin, on donne les 30 valeurs de X qui sont compatibles avec les contraintes bilatérales.

II.5.2 L'écriture des quatre sous-programmes d'entrée de données :

Les quatre sous-programmes qui caractérisent le problème seront écrits comme suit :

```
      FUNCTION F(X)
      COMMON/E2/C1(30),C2(30),NS,ND
      DIMENSION X(1)
      L=NS*ND
      F=0.
      DO 10 J=1,L
   10 F=F-C1(J)*X(J)-C2(J)*X(J)*X(J)
      RETURN
      END

      SUBROUTINE GRADS(X,FP)
      COMMON/E2/C1(30),C2(30),NS,ND
      DIMENSION X(1),FP(1)
      L=NS*ND
      DO 1 J=1,L
    1 FP(J)=-C1(J)-2.*C2(J)*X(J)
      RETURN
      END
```

```
      SUBROUTINE CONT(X,K)
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(30),ILU(42),IVL(30),IVX(42),EPS(2),
     1C(42),G(30,30),U1(30),FP(42),W(30,30),U2(30),U3(30),IP(30),NA
      COMMON/EQ/FY,FZ,FU,DD(42)
      DIMENSION X(1)
      C(1)=X(1)+X(2)+X(3)+X(4)+X(5)+X(6)+X(7)+X(8)+X(9)+X(10)-100.
      DD(1)=D(1)+D(2)+D(3)+D(4)+D(5)+D(6)+D(7)+D(8)+D(9)+D(10)
      C(2)=X(11)+X(12)+X(13)+X(14)+X(15)+X(16)+X(17)+X(18)+X(19)+X(20)-9
     *7.
      DD(2)=D(11)+D(12)+D(13)+D(14)+D(15)+D(16)+D(17)+D(18)+D(19)+D(20)
      C(3)=X(1)+X(11)+X(21)-25.
      DD(3)=D(1)+D(11)+D(21)
      C(4)=X(2)+X(12)+X(22)-40.
      DD(4)=D(2)+D(12)+D(22)
      C(5)=X(3)+X(13)+X(23)-60.
      DD(5)=D(3)+D(13)+D(23)
      C(6)=X(4)+X(14)+X(24)-30.
      DD(6)=D(4)+D(14)+D(24)
      C(7)=X(5)+X(15)+X(25)-20.
      DD(7)=D(5)+D(15)+D(25)
      C(8)=X(6)+X(16)+X(26)-30.
      DD(8)=D(6)+D(16)+D(26)
      C(9)=X(7)+X(17)+X(27)-35.
      DD(9)=D(7)+D(17)+D(27)
      C(10)=X(8)+X(18)+X(28)-30.
      DD(10)=D(8)+D(18)+D(28)
      C(11)=X(9)+X(19)+X(29)-25.
      DD(11)=D(9)+D(19)+D(29)
      C(12)=X(10)+X(20)+X(30)-40.
      DD(12)=D(10)+D(20)+D(30)
      DO 1 I=13,NT
      L=I-12
      C(I)=X(L)
    1 DD(I)=D(L)
      L=0
      DO 2 I=1,NT
      IF(ILU(I).EQ.K) GO TO 2
      L=L+1
      C(L)=C(I)
      DD(L)=DD(I)
    2 CONTINUE
      RETURN
      END



      SUBROUTINE DEPLU(G,ILU)
      DIMENSION G(30,1),ILU(1)
      DO 100 I=1,30
      DO 100 J=1,30
  100 G(I,J)=0.
      I=1
      IF(ILU(1).EQ.0) GO TO 1
      DO 46 J=1,10
   46 G(J,I)=1.
```



```
      I=I+1
    1 IF(ILU(2).EQ.0) GO TO 2
      DO 47 J=11,20
   47 G(J,I)=1.
      I=I+1
    2 K=3
      DO 4 L=1,10
      IF(ILU(K).EQ.0) GO TO 3
      G(L,I)=1.
      L1=L+10
      G(L1,I)=1.
      L2=L1+10
      G(L2,I)=1.
      I=I+1
    3 K=K+1
    4 CONTINUE
      K=13
      DO 6 L=1,30
      IF(ILU(K).EQ.0) GO TO 5
      G(L,I)=1.
      I=I+1
    5 K=K+1
    6 CONTINUE
      RETURN
      END
```

II.5.3 Programme principal dans le cas du problème de transport :

Dans la suite, on donne la partie du programme principal, ainsi les instructions indispensables pour appliquer la deuxième méthode du changement des variables de base CRBAS2.

Les données et les paramètres sont les mêmes qu'en (II - 5) et (II - 5 - 1).

Le programme principal sera écrit comme suit :

```
      COMMON/A/N,NT,M,INCI,IFI,ICL,D(30),ILU(42),IVL(30),IVX(42),EPS(2),
     1C(42),G(30,30),U1(30),FP(42),W(30,30),U2(30),U3(30),IP(30),NA
      COMMON/A1/X(30)
      COMMON/B/FX,H,H1,Y(30),Z(30),U(30)
      COMMON/D1/NVB(12),LT(12,10),P,IS(12)
      COMMON/E0/FY,FZ,FU,DD(42)
      COMMON/E2/C1(30),C2(30),NS,NO
      EQUIVALENCE(NVB1,NVB(1)),(NVB2,NVB(2)),(NVB3,NVB(3)),(NVB4,NVB(4)),
     *,(NVB5,NVB(5)),(NVB6,NVB(6)),(NVB7,NVB(7)),(NVB8,NVB(8)),(NVB9,NVB
     *(9)),(NVB10,NVB(10)),(NVB11,NVB(11)),(NVB12,NVB(12))
    3 FORMAT(25I3)
   13 FORMAT(' IVL=',30I3,9X,I3,5X,I3)
 1000 FORMAT(' ICT=',I3/' X=',10F10.5/' X=',10F10.5/' D=',
     115F8.5/' D=',15F8.5//' FX=',F15.6/' ILU=',42I2/' IVL=',30I2//)
 2000 FORMAT(16F5.2)
 3000 FORMAT('    MAXIMUM OBTENU',//' X OPT.=',15F8.4/8X,15F8.4//'     F
     *OPT.=',F20.8//' NO. DES ITERATIONS =',I4,5X,'NCVB=',I3/' D=',15F8.
     X5/' D=',15F8.5)
 5000 FORMAT('***** PLUS DE CHANGEMENT DE VAR. DE BASE')
C
C     PARAMETRES ET DONNEES DU PROBLEME.
C
      INTEGER P
      NA=30
      NT=42
      N=30
      P=12
      M=P
      NCVB=0
      FY=0.
      FZ=0.
      FU=0.
      EPS(1)=0.01
      EPS(2)=1.E-10
      NS=3
      NO=10
      DO 77 I=1,M
   77 IVX(I)=0
      M1=M+1
      DO 88 I=M1,NT
      J=I-M
   88 IVX(I)=J
      DO 4 I=1,M
    4 ILU(I)=1
      M1=M+1
      DO 5 I=M1,NT
    5 ILU(I)=0
      ICT=0
      READ(105,2000)H,H1
      WRITE(108,2000)H,H1
      READ(105,3)(NVB(J),J=1,M)
      DO 2 I=1,M
      K=NVB(I)
      READ(105,3)(LT(I,J),J=1,K)
```



```
    2 WRITE(108,3)(LT(I,J),J=1,K)
C
C     LECTURE DES COEFFICIENTS C1 ET C2 DE LA FONCTION ECONOMIQUE.
C
      READ(105,2000)(C1(J),J=1,N)
      READ(105,2000)(C2(J),J=1,N)
      WRITE(108,2000)(C1(J),J=1,N)
      WRITE(108,2000)(C2(J),J=1,N)
C
C     LECTURE D'UN VECTEUR X COMPATIBLE AVEC LES CONTRAINTES.
C
      READ(105,2000)(X(J),J=1,N)
      WRITE(108,2000)(X(J),J=1,N)
      M1=NT-N
      DO 6 I=1,N
      IF(X(I).GT.0.0) GO TO 6
      M=M+1
      M2=M1+I
      ILU(M2)=1
    6 CONTINUE
C
C     L'APPLICATION DE LA 2 EME METHODE DU CHANGEMENT DES VARIABLES DE
C     BASE.
C
   11 DO 65 IS1=1,NVB1
      IS(1)=IS1
      DO 65 IS2=1,NVB2
      IS(2)=IS2
      DO 65 IS3=1,NVB3
      IS(3)=IS3
      DO 65 IS4=1,NVB4
      IS(4)=IS4
      DO 65 IS5=1,NVB5
      IS(5)=IS5
      DO 65 IS6=1,NVB6
      IS(6)=IS6
      DO 65 IS7=1,NVB7
      IS(7)=IS7
      DO 65 IS8=1,NVB8
      IS(8)=IS8
      DO 65 IS9=1,NVB9
      IS(9)=IS9
      DO 65 IS10=1,NVB10
      IS(10)=IS10
      DO 65 IS11=1,NVB11
      IS(11)=IS11
      DO 65 IS12=1,NVB12
      IS(12)=IS12
      CALL CRBAS2(NVB,LT,IS,P)
      IF(INCI.NE.0) GO TO 190
C
C     LE NOMBRE DE CHANGEMENT DES VARIABLES DE BASE AUGMENTE D'UN.
C
      NCVB=NCVB+1
    1 CALL GPROJ(P)
      IF(INCI.NE.0) GO TO 100
      FX=F(X)
      IF(IFI.EQ.0) GO TO 10
```

```
      CALL ETAPL
      IF(INCI.NE.0) GO TO 100
      ICT=ICT+1
      WRITE(108,1000)ICT,(X(I),I=1,N),(D(I),I=1,N),FX,(ILU(I),I=1,NT),(I
     1VL(I),I=1,N)
      GO TO 1
   10 FX=F(X)
      WRITE(108,3000)(X(I),I=1,N),FX,ICT,NCVB,(D(I),I=1,N)
      GO TO 19
  100 WRITE(108,13)(IVL(I),I=1,N),INCI,NCVB
   65 CONTINUE
      GO TO 11
  190 WRITE(108,5000)
   19 STOP
      END
```

II - 6 LES RESULTATS

Les résultats qui suivent ont été obtenus, en appliquant l'algorithme du gradient projeté sur le problème de flot énoncé en (II.4.1) ; ainsi qu'au problème de transport énoncé en (II - 5).

Une solution pour le même problème de transport est obtenue en considérant que :

$$\sum_{i=1}^{N} P_i \neq \sum_{j=1}^{M} R_j \quad \text{(cf. I.5.1.b)}.$$

II.6.1 Exemple de problème de flot :

Le problème de flot énoncé en (II.4.1) a été résolu pour différentes valeurs du flot QE.

QE = 25

Itération N°	Contraintes utiles	X_B	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	FX
0			15	10	6	3	6	6	4	6	9	10	
1	1,2,3,4,5,5	2,5,6,7,8,10	13.56	11.44	5.51	4.00	4.05	4.47	6.97	5.51	8.47	11.02	311.90
2	"	"	11.79	13.21	4.57	6.28	0.93	1.07	12.14	4.57	7.35	13.07	296.70
3	1,2,3,4,5,5, 21,22	1,2,3,4,5,6, 7,10	11.78	13.22	4.37	7.41	0.00	0.00	13.22	4.37	7.41	13.22	293.28
4	"	"	11.34	13.66	3.88	7.45	0.00	0.00	13.66	3.88	7.45	13.66	292.32
5	"	"	10.57	14.43	2.98	7.58	0.00	0.00	14.43	2.98	7.58	14.43	290.85
6	1,2,3,4,5,5, 13,22	1,2,3,4,5,6, 7,9	10.00	15.00	2.40	7.59	0.002	0.00	15.00	2.40	7.59	15.00	290.05
7	"	"	10.00	15.00	2.38	7.61	0.007	0.00	15.00	2.38	7.61	15.01	290.03
8	"	"	10.00	15.00	1.06	8.63	0.312	0.00	15.00	1.06	8.63	15.31	289.14
9	1,2,3,4,5,5, 13,22,24	1,2,3,....,9	10.00	15.00	0.00	9.67	0.33	0.00	15.00	0.00	9.67	15.33	288.511
10	"	"	10.00	15.00	0.00	9.68	0.32	0.00	15.00	0.00	9.68	15.32	288.510
11	"	"	10.00	15.00	0.00	9.71	0.29	0.00	15.00	0.00	9.71	15.29	288.508
12	"	"	10.00	15.00	0.00	9.84	0.16	0.00	15.00	0.00	9.84	15.16	288.502

Nombre des changements de base = 4

QE = 45

Itération N°	Contraintes utiles	X_B	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	FX
0			28	17.	10.	8.	10.	10.	7.	10.	18.	17.	
1	1,2,3,4,5,6	2,5,6,7,8,10	23.52	21.48	9.39	7.54	6.60	9.50	11.97	9.39	17.04	18.57	632.71
2	1,2,3,4,5,6, 13	1,2,5,6,7,8, 10	20.78	24.22	9.28	7.56	3.94	9.22	15.00	9.28	16.78	18.94	617.47
3	"	"	19.26	25.74	8.61	6.16	4.48	10.74	15.00	8.61	16.91	19.48	616.86
4	"	"	19.76	25.24	8.26	5.95	5.55	10.24	15.00	8.26	16.19	20.55	616.71
5	"	"	19.36	25.64	7.96	5.65	5.74	10.64	15.00	7.96	16.3	20.74	616.66
6	"	"	19.55	25.45	7.73	5.72	6.1	10.45	15.00	7.73	16.17	21.1	616.65
7	"	"	19.4	25.6	7.58	5.67	6.15	10.6	15.00	7.58	16.27	21.15	616.64

Nombre des changements de base = 3

QE = 60

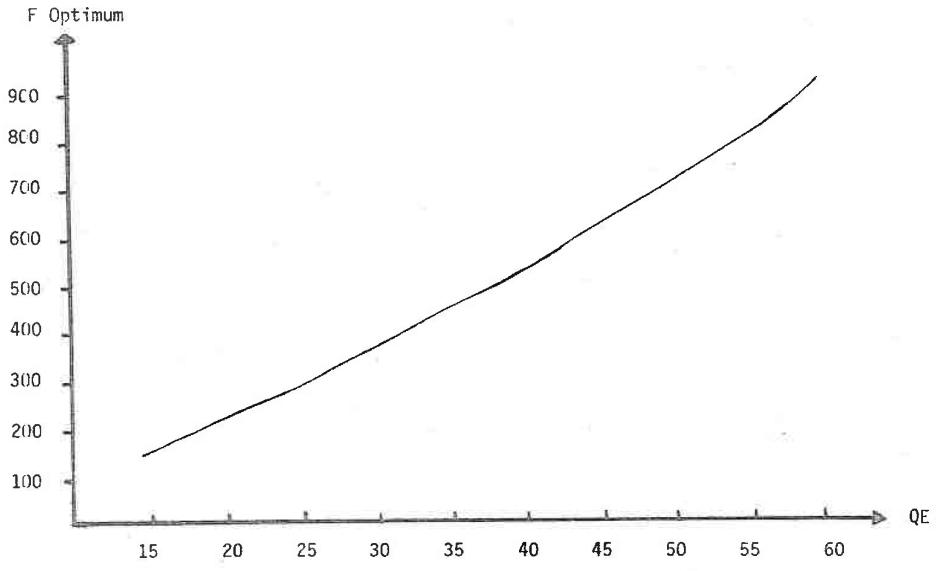
Itération N°	Contraintes utiles	X_B	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	FX
0			34.	26.	13.	8.	13.	15.	11.	13.	23.	24.	
1	1,2,3,4,5,6	2,5,6,7,8,10	31.29	28.71	12.73	7.55	11.02	14.82	13.89	12.73	22.37	24.9	921.72
2	1,2,3,4,5,6, 8,16	1,2,3,5,6,7, 8,10	30.00	30.00	12.84	7.47	9.7	14.7	15.3	12.84	22.16	25.00	910.41
3	1,2,3,4,5,6, 8,13,16	1,2,3,4,5,6, 7,8,10	30.00	30.00	12.97	7.03	10.00	15.00	15.00	12.97	22.03	25.00	911.65
4	1,2,3,4,5,6, 8,13	1,2,3,4,5,6, 7,10	30.00	30.00	13.04	6.96	10.00	15.00	15.00	13.04	21.96	24.99	911.62
5	"	"	30.00	30.00	13.18	6.83	9.99	15.00	15.00	13.18	21.83	24.99	911.56
6	"	"	30.00	30.00	13.46	6.58	9.97	15.00	15.00	13.46	21.58	24.97	911.46
7	"	"	30.00	30.00	13.97	6.12	9.91	15.00	15.00	13.97	21.12	24.91	911.28
8	"	"	30.00	30.00	14.89	5.35	9.76	15.00	15.00	14.89	20.35	24.76	911.03
9	1,2,3,4,5,6, 8,9,13	1,2,3,4,5,6, 7,8,10	30.00	30.00	15.00	5.34	9.66	15.00	15.00	15.00	20.34	24.66	911.01
10	"	"	30.00	30.00	15.00	5.32	9.68	15.00	15.00	15.00	20.32	24.68	911.008
11	"	"	30.00	30.00	15.00	5.28	9.72	15.00	15.00	15.00	20.28	24.72	911.006
12	"	"	30.00	30.00	15.00	5.18	9.82	15.00	15.00	15.00	20.18	24.82	911.002

Nombre des changements de base = 3

Nous trouverons au tableau suivant les valeurs de la fonction objectif optimale pour les valeurs de QE de 15 à 60 de 5 en 5.

QE	F optimum	Nombre des itérations
15	163.589	9
20	224.637	13
25	288.502	12
30	362.223	13
35	443.490	11
40	528.506	10
45	616.637	7
50	707.711	7
55	804.110	11
60	911.002	12

Ces résultats apparaissent également sur la courbe suivante


F Optimum
900
800
700
600
500
400
300
200
100
15
20
25
30
35
40
45
50
55
60
QE

II.6.2 Les résultats de l'exemple du problème de transport

Les résultats suivants ont obtenus pour le problème traité en [1]. Chaque tableau représente les résultats d'une "itération" du calcul. On donne quelques résultats intermédiaires entre la 1ère itération et la dernière.

ITER 1

x		x		x	
1	10.5867	11	8.8067	21	5.6066
2	19.7865	12	3.4067	22	16.8067
3	37.5867	13	18.8065	23	3.6067
4	6.5199	14	9.3401	24	14.1400
5	1.6533	15	9.6732	25	8.4734
6	1.6533	16	15.8734	26	12.4733
7	10.7867	17	3.8067	27	20.4066
8	3.7867	18	4.8067	28	21.4066
9	2.8534	19	13.2734	29	8.8733
10	4.5867	20	9.2067	30	26.2066

Variables de base :

X_B : x_1, x_2, x_3, x_{13}, x_{14}, x_{15}, x_{16}, x_{21}, x_{27}, x_{28}, x_{29}, x_{30}.

Contraintes "utiles"

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12.

Valeur de la fonction objectif :

$F(X^1) = 1204.6188.$

ITER 10

1	13.1994	11	9.7950	21	2.0056
2	20.1974	12	0.8313	22	18.9713
3	38.3082	13	21.6918	23	0.000
4	7.7708	14	2.8727	24	19.3565
5	4.2221	15	8.6483	25	7.1296
6	0.000	16	11.6172	26	18.3828
7	12.0933	17	3.3903	27	19.5164
8	0.000	18	13.4936	28	16.5064
9	0.000	19	16.6090	29	8.3910
10	4.2089	20	8.0508	30	27.7403

Variables de base :

X_B : x_1, x_2, x_3, x_6, x_8, x_9, x_{13}, x_{14}, x_{15}, x_{16}, x_{20}, x_{22}, x_{27}, x_{28}, x_{29}, x_{30}.

Contraintes "utiles" :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 17, 19, 20, 35.

Valeur de la fonction objectif :

$F(X^{10}) = 1106.1888.$

ITER 20

1	13.8490	11	11.1510	21	0.000
2	19.5360	12	0.000	22	20.4640
3	30.2509	13	29.7491	23	0.000
4	1.5056	14	0.000	24	28.4944
5	10.9195	15	1.7557	25	7.3249
6	0.000	16	0.000	26	30.000
7	20.1827	17	2.8478	27	11.9695
8	0.000	18	26.4233	28	3.5767
9	0.000	19	25.000	29	0.000
10	3.7564	20	0.0731	30	36.1705

Variables de base :

X_B : x_1, x_2, x_3, x_4, x_6, x_8, x_9, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{18}, x_{19}, x_{20}, x_{21}, x_{23}, x_{24}, x_{26}, x_{27}, x_{29}.

Contraintes "utiles" :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 18, 20, 21, 24, 26, 28, 33, 35, 39.

Valeur de la fonction objectif :

$F(X^{20}) = 1016.0172.$

ITER 32

1	13.3352	11	9.3424	21	2.3224
2	17.8832	12	0.000	22	22.1168
3	23.4246	13	36.5754	23	0.000
4	0.000	14	0.000	24	30.000
5	11.7786	15	0.000	25	8.2214
6	0.000	16	0.000	26	30.000
7	33.5785	17	1.4215	27	0.000
8	0.000	18	24.6607	28	5.3393
9	0.000	19	25.000	29	0.000
10	0.000	20	0.000	30	40.000

Variables de base :

X_B : x_1, x_2, x_3, x_4, x_5, x_6, x_8, x_9, x_{10}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{19}, x_{20}, x_{21}, x_{23}, x_{24}, x_{26}, x_{27}, x_{28}, x_{29}, x_{30}.

Contraintes utiles :

1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21, 23, 24, 26, 27, 28, 29, 30.

Valeur de la fonction objectif :

$F(X^{32}) = 1005.16.$

II.6.3 Comparaison avec la méthode de Bellman [1]

1) L'essentiel de la méthode de Bellman est que cette méthode est une suite itérative d'étapes, à chaque étape on se ramène à un problème à deux points de distribution (les flots relatifs aux points de distribution restent bloqués) ; problème qui se résoud simplement par programmation dynamique. Dans cette phase Bellman s'est limité à des inconnues à valeurs entières.

2) Nous avons repris le même problème par la méthode du gradient projeté, nos inconnues peuvent alors prendre des valeurs quelconques, non entières.

Les résultats des deux méthodes apparaissent dans le tableau ci-dessous, en notant par P.D. pour les résultats obtenus par la méthode de programmation dynamique, et par G.P. pour la méthode de gradient projeté.

x	P.D	G.P.	x	P.D.	G.P.	x	P.D.	G.P.
1	13	13.34	11	9	9.34	21	3	2.32
2	18	17.88	12	0	0.00	22	22	22.12
3	24	23.42	13	36	36.58	23	0	0.00
4	0	0.00	14	0	0.00	24	30	30.00
5	12	11.78	15	0	0.00	25	8	8.22
6	0	0.00	16	0	0.00	26	30	30.00
7	33	33.58	17	2	1.42	27	0	0.00
8	0	0.00	18	25	24.66	28	5	5.34
9	0	0.00	19	25	25.00	29	0	0.00
10	0	0.00	20	0	0.00	30	40	40.00

Valeur de la fonction objectif obtenue par (P.D.)

= 1005.26.

Valeur de la fonction objectif obtenue par (G.P.)

= 1005.16.

On remarque la bonne concordance des résultats ; en général la valeur obtenue par Bellman est l'entier le plus proche de notre solution.

Les valeurs à l'optimum de la fonction objectif sont également voisines, le minimum obtenu par Bellman étant très légèrement supérieur au nôtre, ceci étant conséquence des contraintes introduites implicitement par Bellman en supposant les variables entières.

Notons enfin que la méthode du gradient projeté permet de traiter un nombre quelconque de points de distribution, alors que la méthode d'approximation successive de Bellman, pour laquelle nous n'avons aucun théorème de convergence devient extrêmement lourde si le nombre de points de distribution dépasse trois.

II.6.4 Exemple de problème de transport (cas où l'offre dépasse la demande)

Dans la suite, on donne les résultats de l'exemple énoncé en (II-5), en considérant que :

$$\sum_{i=1}^{3} P_i > \sum_{j=1}^{10} R_j$$

On considère que :

$$\sum_{i=1}^{10} x_i \leq 110 ;$$

$$\sum_{i=11}^{20} x_i \leq 100 ;$$

$$\sum_{i=21}^{30} x_i \leq 145 ;$$

et la solution initiale était choisie comme suit :

8	15	20	10	5	10	12	5	5	15	≤ 110
7	10	20	10	5	10	8	10	5	10	≤ 100
10	15	20	10	10	10	15	15	15	15	≤ 145
25	40	60	30	20	30	35	30	25	40	

On considère aussi que les autres paramètres et données ont été les mêmes que dans (II - 5). Les tableaux suivants sont des résultats obtenus entre la première itération et la dernière.

ITER 1

1	8.5916	11	7.4198	21	8.9886
2	15.1336	12	9.9618	22	14.9046
3	20.8206	13	21.5649	23	17.6145
4	10.2099	14	9.2439	24	10.4962
5	5.3053	15	5.5344	25	9.1603
6	8.9695	16	10.2290	26	10.8015
7	12.4199	17	7.7328	27	14.8473
8	4.2366	18	11.5267	28	14.2367
9	3.8550	19	6.4313	29	14.7137
10	14.8473	20	10.3054	30	14.8473

Variables de base :

X_B : x_1, x_2, x_3, x_4, x_5, x_8, x_9, x_{10}, x_{13}, x_{16}, x_{17}.

Contraintes "utiles" :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12.

Valeur de la fonction objectif :

$F(X^1) = 1290.3207.$

ITER 15

1	12.9641	11	10.0385	21	1.9975
2	19.0394	12	1.1570	22	19.8036
3	27.3751	13	32.6249	23	0.000
4	9.6105	14	0.000	24	20.3895
5	8.6934	15	4.1644	25	7.1422
6	0.000	16	0.3356	26	29.6644
7	18.1322	17	3.4594	27	13.4084
8	0.000	18	23.6965	28	6.3035
9	0.000	19	17.5059	29	7.4941
10	14.1853	20	7.0177	30	18.7970

Variables de base :

X_B : x_1, x_2, x_4, x_5, x_6, x_8, x_9, x_{10}, x_{11}, x_{13}, x_{14}, x_{16}, x_{17}, x_{18}, x_{19}, x_{22}, x_{23}.

Contraintes "utiles"

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 19, 21, 22, 27, 36.

Valeur de la fonction objectif

$F(X^{15}) = 1038.5037.$

ITER 30

1	13.9654	11	11.0346	21	0.000
2	19.2738	12	0.000	22	20.7262
3	25.8139	13	34.1861	23	0.000
4	0.9020	14	0.000	24	29.0980
5	11.8500	15	0.5317	25	7.6183
6	0.000	16	0.000	26	30.000
7	24.3510	17	2.7534	27	7.8956
8	0.000	18	26.4941	28	3.5059
9	0.000	19	25.000	29	0.000
10	13.8454	20	0.0640	30	26.0906

Variables de base :

X_B : x_1, x_2, x_3, x_4, x_5, x_6, x_8, x_9, x_{11}, x_{12}, x_{13}, x_{14}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{26}, x_{29}.

Contraintes "utiles" :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 19, 21, 22, 25, 27, 29, 33, 34, 36, 42.

Valeur de la fonction objectif :

$F(X^{30}) = 1006.797.$

ITER 45

1	13.9198	11	10.7688	21	0.3114
2	18.8365	12	0.000	22	21.1635
3	24.4279	13	35.5721	23	0.000
4	0.000	14	0.000	24	30.000
5	12.1049	15	0.000	25	7.8951
6	0.000	16	0.000	26	30.000
7	32.4804	17	2.3911	27	0.1285
8	0.000	18	26.2680	28	3.7320
9	0.000	19	25.000	29	0.000
10	8.2304	20	0.000	30	31.7696

Variables de base :

X_B : x_2, x_3, x_4, x_5, x_6, x_8, x_9, x_{11}, x_{12}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18} x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{26}, x_{29}, x_{30}.

Contraintes "utiles" :

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 17, 19, 21, 22, 25, 27, 28, 29, 33, 36, 42.

Valeur de la fonction objectif :

$F(X^{45}) = 998.7198.$

ITER 63

1	14.2051	11	10.7949	21	0.000
2	21.0015	12	0.000	22	18.9985
3	25.0684	13	34.9316	23	0.000
4	0.000	14	0.000	24	30.000
5	13.000	15	0.000	25	7.000
6	0.000	16	0.000	26	30.000
7	33.5625	17	1.4375	27	0.000
8	0.000	18	27.8359	28	2.1641
9	0.000	19	25.000	29	0.000
10	2.4788	20	0.000	30	37.5212

Variables de base :

X_B : x_1, x_4, x_5, x_6, x_8, x_9, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{26}, x_{27}, x_{29}, x_{30}.

Contraintes "utiles" :

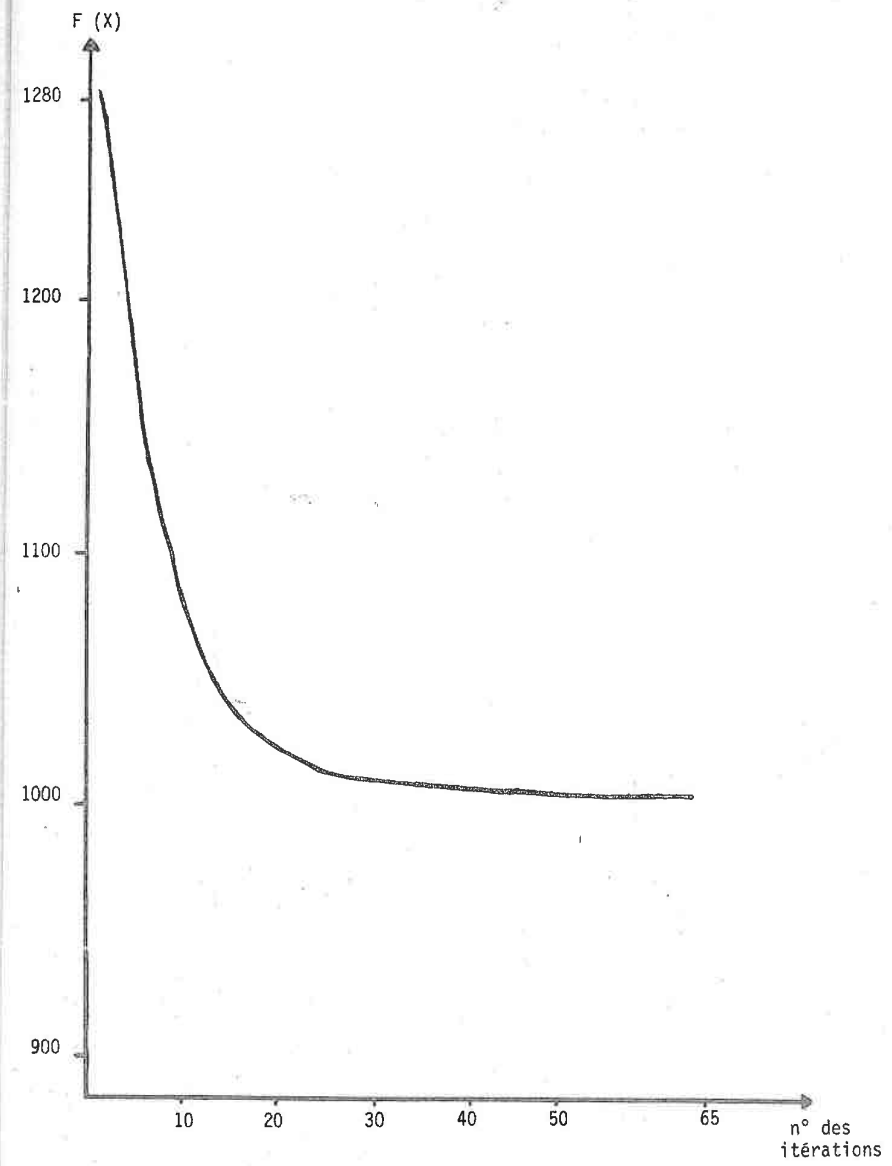
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 17, 19, 21, 22, 25, 27, 28, 29, 33, 34, 36, 40, 42.

Valeur de la fonction objectif :

$F(X^{63}) = 997.3477.$

La convergence de cette méthode apparaîtra clairement en traçant la valeur de la fonction objectif en fonction des itérations correspondantes.

Nombre des itérations	F (X)
1	1290.32
5	1167.77
10	1077.76
15	1038.50
20	1024.15
25	1011.31
30	1006.80
35	1003.92
40	1001.29
45	998.72
50	998.08
55	997.36
60	997.35
63	997.348


F (X)
1280
1200
1100
1000
900
10
20
30
40
50
65
n° des
itérations

CONCLUSION

Compte tenu du caractère d'unimodularité de la matrice des contraintes, il nous a donc été possible d'adapter la méthode du gradient projeté et d'écrire un programme d'optimisation efficace, qui permet par exemple de traiter un problème de quarante variables sur un ordinateur de 32 K mots de mémoire centrale ; le temps correspondant à une itération est alors de cinq à sept secondes sur un Mitra 125 à virgule flottante micro-programmée.

BIBLIOGRAPHIE

[1] BELLMAN R. and DREYFUS S.
"Applied dynamic programming"
(Princeton Univ. Press, 1962).

[2] BERGE C. "Théorie des graphes et ses applications"
(Dunod, 1963).

[3] BERGE C. "Graphes et hypergraphes".
(Dunod, 1970).

[4] CAMION P. "Characterization of totally unimodular matrices"
(Proc. of the A.M. Soc. 1964, pp. 1068-1073).

[5] CAMION P. "Characterisation des matrices unimodulaires"
(Cahiers du C.E.R.O., Vol. 5, n°4, 1963, pp. 181 - 190).

[6] CEDERBAUM I. "Matrices all of whose elements and determinants are 1, -1, 0"
(J. Math. and Phys. 36, 1958, pp. 351 - 361).

[7] COMPOINT P. "Les graphes en recherche opérationnelle"
(Dunod, 1972).

[8] FAURE R. "Graphes et applications"
(Cours polycopié de la faculté des Sciences de PARIS VI).

[9] GHOUILA - HOURI A.
"Characterisation des matrices totalement unimodulaires"
(C.R. Acad. Sc. 254 (1962), pp. 1192).

[10] GHOUILA - HOURI A.
"Flots et Tensions dans un graphe"
(Ann. Ec. Norm., (3), LXXXI Fasc. 3, pp. 285).

[11] HELMER J. Y. "La commande optimale en économie"
(Dunod 1972).

[12] KAUFMANN A. "Méthodes et modèles de la Recherche Opérationnelle"
(Dunod, 1968, Tome 2).

[13] KORGANOFF A. "Méthodes de calcul numérique"
(Dunod, 1961, Vol. 1).

[14] LEGRAS J. "Méthodes et techniques de l'analyse numérique"
(Dunod, 1971).

[15] LEGRAS J. "Algorithmes d'optimisation - Problèmes avec contraintes et contrôle optimal"
(Cours polycopié de la Faculté des Sciences de NANCY I).

[16] LEGRAS J. "Algorithmes et programmes d'optimisation non linéaire avec contraintes - Application au contrôle optimal"
(MASSON - En cours de Parution).

[17] LUCAS PUN "Introduction à la pratique de l'optimisation".
(Dunod, 1972).

[18] PALLU DE LA BARRIERE
"Cours d'automatique théorique"
(Dunod, 1966).

[19] YOUSSEF M. Y.
"Problèmes de tournées de ramassage lorsque les sources produisent des quantités variables en fonction du temps"
(Thèse 3ème Cycle - Université de Paris VI, 1974)

NOM DE L'ETUDIANT : Monsieur YOUSSEF Mohamed Yehia

NATURE DE LA THESE : DOCTORAT ES SCIENCES

VU, APPROUVE

et PERMIS D'IMPRIMER

NANCY LE 29 SEP 1978 6656

LE PRESIDENT DE L'UNIVERSITE DE NANCY I