METHODES AVANCEES

1- RELAXATION LAGRANGIENNE

1.1 Défintion

Considérons un problème linéaire dont la fonction objectif est notée *f*(*x*).

1. Si l'on remonte dans cette fonction objectif une contrainte de la forme $g^J(x) \geq C_J$, la nouvelle fonction objectif devient :
    a. $f(x) + \lambda.(C^J - g_J(x))$ s'il s'agit d'une minimisation
    b. $f(x) - \lambda.(C^J - g_J(x))$ s'il s'agit d'une maximisation.

   Quant à la mise à jour de λ, elle est de la forme $\lambda^{k+1} = \max\left(0; \lambda^k + \frac{C_J - g^J(x_i)}{k}\right)$.

2. Si l'on remonte dans cette fonction objectif une contrainte de la forme $h^J(x) \leq C_J$, la nouvelle fonction objectif devient :
    c. $f(x) + \lambda.(h_J(x) - C^J)$ s'il s'agit d'une minimisation
    d. $f(x) - \lambda.(h_J(x) - C^J)$ s'il s'agit d'une maximisation.

   Quant à la mise à jour de λ, elle est de la forme $\lambda^{k+1} = \max\left(0; \lambda^k + \frac{h^J(x_i) - C_J}{k}\right)$.

3. Si on considère une contrainte de la forme $q^J(x) = C_J$, on réécrit cette contrainte comme $q^J(x) \leq C_J$ et $q^J(x) \geq C_J$, et la nouvelle fonction objectif devient :
    e. $f(x) + \lambda^1\left(C_J - q^J(x)\right) + \lambda^2\left(q^J(x) - C_J\right)$ s'il s'agit d'une minimisation
    f. $f(x) + \lambda^2\left(C_J - q^J(x)\right) + \lambda^1\left(q^J(x) - C_J\right)$ s'il s'agit d'une maximisation

L'important est que **lorsque la contrainte est violée, le coefficient λ augmente,** alors que si la **contrainte est satisfaite, il décroît** (en restant positif ou nul) pour éventuellement devenir nul. Si la contrainte est violée, dans un problème de minimisation, un terme positif s'ajoute à la fonction objectif, alors que dans le cas d'une maximisation, il s'en retranche.

1.2. Justification de la méthode

Dans le paragraphe précédent nous avons montré que la méthode de la relaxation lagrangienne semble fonctionner sur quatre exemples et nous avons proposé une mise à jour des coefficients sans justifier la formule ni présenter l'idée sous jacente.

L'objectif de ce paragraphe est montrer pourquoi cette méthode fonctionne et fournit dans le cas d'un problème de minimisation une borne inférieure et dans le cas d'un problème de maximisation une borne supérieure du problème de départ. De plus, nous montrerons différentes techniques de mise à jour des coefficients, point qui semble important, à la vue du nombre d'itérations nécessaires sur les « petits » exemples précédents.

1.3. Pourquoi la relaxation lagrangienne ?

La relaxation lagrangienne est une technique qui permet d'ajouter à la fonction économique une pénalité d'autant plus élevée que la contrainte est moins respectée. Au lieu d'exclure les points qui violent les contraintes, la méthode permet de les envisager. Comme toutes les méthodes de pénalités, la méthode de relaxation lagrangienne dépend d'un paramètre qui permet de régler l'impact de la violation. Quand la pénalité devient très grande devant la fonction objectif on tend vers l'optimum du problème initial. Remarquons toutefois que la fonction à optimiser devient de plus en plus mal conditionnée.

Il existe deux grandes classes de méthodes par pénalités :

- la première regroupe les méthodes telles que la fonction de pénalité devient nulle dans le domaine faisable. C'est une méthode extérieure. L'optimum pénalisé n'est pas faisable.
- la deuxième regroupe les fonctions telles que la fonction devient infinie quand on sort du domaine faisable. L'optimum est faisable mais la méthode a besoin d'un point de départ faisable.

Les méthodes **extérieures** consistent à remplacer le problème

$$\min\ f(x)$$

$$\forall i = 1..n,\ g_i(x) \geq b_i$$

$$\forall i = 1..m,\ s_i(x) \geq b_i$$

$$x_i \in IN$$

Par une suite de problème P_k

$$\min\ f(x) + S_k . \sum_{i=1}^{n} h\left(g_i(x)\right)$$

$$\forall i = 1..n,\ g_i(x) \geq b_i$$

$$\forall i = 1..m,\ s_i(x) \geq b_i$$

$$x_i \in IN$$

Où les S_k forment une suite croissante tendant vers l'infini. On note $H(x) = \sum_{i=1}^{n} h\left(g_i(x)\right)$ et x_k la solution du problème P_k.

1.4. La relaxation lagrangienne est une méthode de pénalités

La méthode de Lagrange consiste à choisir (pour le sous-problème P_k), la fonction :

$\min L(\lambda, x) = f(x) + \lambda . g(x)$ avec un vecteur λ ayant toutes ses composantes supérieures ou égales à zéro, ce que nous notons $\lambda \geq 0$. Les composantes de g sont notées g_i.

On cherche alors à trouver un minimum global de la fonction $L(\lambda, x)$ qui dépend des vecteurs λ et x. Ce minimum s'appelle un point-col et est noté (λ^*, x^*). Il est possible ensuite de montrer que x^* est aussi un minimum global de $f(x)$.

La fonction de Lagrange $L(\lambda, x) = f(x) + \lambda.g(x)$ est une méthode de pénalités avec :

- $g(x) \geq 0$ dans la région infaisable. Donc le second terme $\lambda.g(x)$ augmente la valeur de la fonction $L(\lambda, x)$ lorsqu'on recherche le minimum.
- $g(x) \leq 0$ dans la région faisable ce qui diminuerait artificiellement la valeur du minimum. C'est la raison pour laquelle on impose que $\forall i, \lambda_i.g_i(x^*) = 0$.

1.5. Considérations de convergence

Les méthodes extérieures consistent à prendre comme fonction de pénalité une fonction qui tend vers l'infini au voisinage de 0 par exemple $\lambda^i = \frac{1}{i}$.

Considérons le problème suivant : trouver x^* minimisant $c^t x$ sous les contraintes $x \geq 0$ et $Ax - b \geq 0$.

Pour $x \geq 0$, définissons la fonction de Lagrange de paramètre λ (avec $\lambda \geq 0$) par $L(x, \lambda) = c^t x + \lambda^t (b - Ax)$.

Si x^* vérifie $x^* \geq 0$ et $Ax^* - b \geq 0$ et minimise $c^t.x^*$, dès que $\lambda \geq 0$, on a vu au chapitre 1 que :

$$c^t.x^* = \inf_x \left\{ c^t x \middle| x \geq 0, Ax - b \geq 0 \right\} = \inf_x \sup_\mu \left\{ c^t x + \mu^t b - \mu^t Ax \middle| x \geq 0, \mu \geq 0 \right\}$$

$$\geq \sup_\mu \inf_x \left\{ c^t x + \mu^t b - \mu^t Ax \middle| x \geq 0, \mu \geq 0 \right\} \geq \inf_x \left\{ c^t x + \lambda^t b - \lambda^t Ax \middle| x \geq 0 \right\} = \inf_x \left\{ L(x, \lambda) \middle| x \geq 0 \right\}.$$

Ceci montre que, pour un problème de minimisation, le minimum (en *x*) atteint par la fonction de Lagrange *L*(*x*,λ) pour un $\lambda \geq 0$ fixé, est un minorant de $c^t.x^*$. Ceci reste valable sous la contrainte supplémentaire : *x* est un vecteur d'entiers (minimisation en nombres entiers). De même, pour un problème de maximisation (éventuellement en nombres entiers), le maximum (en *x*) atteint par la fonction de Lagrange *L*(*x*,λ) pour un $\lambda \geq 0$ fixé, est un majorant de $c^t.x^*$.

Ces propriétés, démontrées ici lorsque **toutes** les contraintes sont passées en fonction objectif, restent vraies lorsque seulement **certaines** contraintes sont passées en fonction objectif.

2- GENERATION DE COLONNES

De manière générale, la technique de génération de colonnes est adaptée à la résolution de problèmes comportant peu de contraintes, mais un grand nombre de variables.

On s'intéresse donc à des problèmes de la forme : $A.x = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ & A_{21} & & \\ & & \cdots & \\ & & & A_{pn} \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ \\ x_n \end{pmatrix} \geq \begin{pmatrix} b_1 \\ b_2 \\ \\ b_p \end{pmatrix}$ avec *n* suffisamment grand par rapport à *p*.

Le principe consiste à résoudre un Programme Maître Restreint (PMR) qui se limite à *k* colonnes c'est-à-dire ne comportant que *k* variables parmi toutes celles possibles. Ensuite, on introduit progressivement certaines des variables (colonnes) inutilisées afin d'améliorer la qualité de la solution obtenue. Pour choisir parmi les variables inutilisées celle que l'on va introduire, on s'appuie

sur le coût réduit de chaque variable inutilisée et l'on introduit (pour un problème de minimisation) celle qui a le coût réduit minimal. Lorsque les coûts réduits des variables inutilisées sont tous positifs ou nuls, l'optimum est atteint. On obtient ainsi un schéma itératif (figure erreur ! **il n'y a pas de texte répondant à ce style dans ce document.**–1).

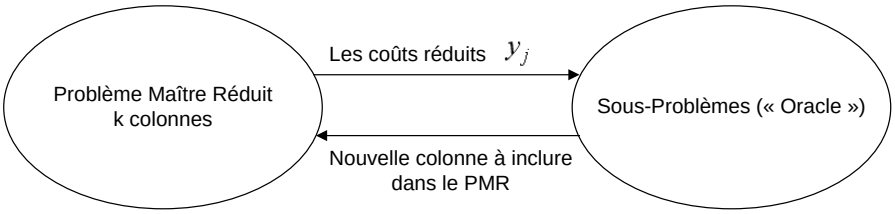

Figure ***Erreur ! Il n'y a pas de texte répondant à ce style dans ce document.****–1. Principe de la génération de colonnes*

Nous présentons ci-dessous la méthode sur quelques exemples illustratifs.

3- DANTZIG-WOLFE

Principes

La méthode de Dantzig-Wolfe consiste à supprimer des variables d'un problème tout en conservant l'ensemble des contraintes. Elle s'applique aux programmes linéaires ayant au moins une contrainte couplante. Une contrainte est couplante si elle fait intervenir la totalité (ou la quasi-totalité) des variables du problème. La méthode de Danzig-Wolfe est une méthode de génération de colonnes particulière comme cela apparaîtra clairement plus loin.

Considérons le problème suivant :

Minimiser $c^T.x$

sous les contraintes

$$A.x = b$$

$$x \geq 0$$

Pour simplifier supposons que la contrainte couplante soit la contrainte numéro 1. La matrice est de la forme suivante, où les A_{ij} sont des matrices et les a_{ij} des réels :

$$A.x = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1K-1} & a_{1K} \\ & a_{22} & & & & \\ & & a_{33} & & & \\ & & & \dots & & \\ & & & & a_{K-2K} & \\ & & & & & a_{KK} \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_K \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_K \end{pmatrix}$$

L'idée consiste à décomposer le problème de départ en un sous problème maître et N sous-problèmes.

Le problème maître comprend :

- la fonction objectif initiale ;
- la contrainte couplante ;
- *N* contraintes de convexité.

Le sous-problème i comprend :

- la fonction objectif restreinte à un certain nombre de variables ;
- les contraintes où apparaissent seulement les variables de la fonction objectif à l'exclusion des autres.

A titre d'exemple, le programme linéaire ci-dessous peut être décomposé en 2 sous-problèmes et en un programme maître en considérant que C1 est la contrainte couplante, qu'un premier bloc peut être fait avec x_2 et x_3 et le deuxième avec x_1.

$\max 7.x_1 + 5.x_2 + 3.x_3$
$C1: \boxed{x_1 + 2x_2 + x_3 \le 10}$ contrainte couplante
$C2: \boxed{x_2 + x_3 \le 5}$
$C3: \boxed{2x_2 + x_3 \le 8}$ Bloc 1
$C4: \boxed{x_1 \le 3}$ Bloc 2
$C5: x_1 \ge 0, x_2 \ge 0, x_3 \ge 0$

Finalement, la décomposition obtenue est la suivante :

$\max 7.x_1 + 5.x_2 + 3.x_3$	$\max 5.x_2 + 3.x_3$	$\max 7.x_1$
$C1: x_1 + 2x_2 + x_3 \le 10,$	$C1: x_2 + x_3 \le 5,$	$C1: x_1 \le 3,$
$C2:$	$C2: 2x_2 + x_3 \le 8,$	$C2: x_1 \ge 0.$
$C3:$	$C3: x_2 \ge 0, x_3 \ge 0,$	
$C5: x_1 \ge 0, x_2 \ge 0, x_3 \ge 0,$		
Le programme maître	le sous-problème 1	le sous-problème 2

Le problème maître est exprimé en fonction de variables auxiliaires généralement notées λ_i. Les variables x_i sont exprimées en fonction des vecteurs v_{ij}, où v_{ij} est le $j^{ième}$ vecteur du sous-problème *i*.

La première étape de la méthode consiste à rechercher les différents vecteurs v_{ij} de chaque sous-problème. Soit B_i la base du $i^{ème}$ sous-problème : $B_i = \{v_{ij}\}$. La base B_i est un ensemble de vecteurs v_{ij}, chacun ayant une dimension égale au nombre de variables du sous-problème *i*. La deuxième étape consiste à exprimer les variables x_i du sous-problème *i* en fonction de λ_i dans la base B_i du sous-problème. Pour chaque sous-problème (bloc), on ajoute une contrainte de convexité au programme maître. Ensuite il s'agit d'un processus itératif pendant lequel on résout successivement le programme maître et les N sous-problèmes.

Si la résolution d'un sous-problème conduit à une solution de coût inférieur au coût réduit de la contrainte de convexité du sous-problème, alors on met à jour la base du sous-problème. Par contre, si aucun sous-problème ne conduit à une solution de coût inférieur au coût réduit de la contrainte de convexité, le processus itératif s'arrête. La solution optimale est la solution du dernier programme maître résolu.

La méthode vient d'une relaxation lagrangienne cachée dans le choix de la prochaine colonne à introduire. En effet si on remonte la contrainte couplante dans la fonction objectif on obtient comme fonction objectif : $\sum_j c_{ij}.x_{ij} - \pi\left(b_i - \sum_j a_{ij}.x_{ij}\right)$ où π est le coefficient de mise à jour. Notons que si on a plusieurs contraintes couplantes on peut obtenir une expression légèrement différente de la forme : $\sum_j c_{ij}.x_{ij} - \sum_i \pi_i\left(b_i - \sum_j a_{ij}.x_{ij}\right)$.

L'expression $\sum_j c_{ij}.x_{ij} - \pi\left(b_i - \sum_j a_{ij}.x_{ij}\right)$ peut se réécrire $\sum_j (c_{ij} - \pi.a_{ij})x_{ij} - \pi.b_i$. Ainsi pour savoir si la solution d'un sous-problème doit faire l'objet d'un ajout dans le programme maître, il suffit de vérifier que $\sum_j (c_{ij} - \pi.a_{ij})x_{ij} - \pi.b_i$ soit bien négatif.

Algorithme de principe :

Calculer les bases B_i de chaque sous-problème.

iter := 0

Repeter

iter := iter + 1

Exprimer les variables x_i en fonction des λ_i et des v_{ij}

Ajouter dans le programme maître *N* contraintes de convexité Cv_i pour $i = 1..N$

Résoudre le programme maître exprimé en fonction des λ_i et v_{ij}

π_1 = variable duale de la contrainte de couplage

σ_i = variable duale de la i^{ieme} contrainte de convexité

Stop :=true

Pour $i := 1$ **à** N **faire**

Calculer les coefficients de la fonction objectif de chaque sous-problème

selon la formule : $c_i^{iter} = c_i - \pi_1.a_i$ où a_i est

le coefficient de la variable x_i dans la contrainte couplante du

problème initial.

Résoudre le sous-problème *i*

Soit Z^i la solution optimale et x_i^* les variables x_i à l'optimum

Si $Z^i < \sigma_i$ **alors**

// une amélioration est possible à partir du sous-problème *i*

Soit v_{ij}^* le vecteur tel que $v_{ij}^*(k) = x_i^*$

Ajouter v_{ij}^* à la base B_i .

Mettre à jour la contrainte couplante que si la solution est bornée

Stop :=false

Fin Si

Fin Pour

Jusqu'à ce que (stop=true).

Une attention particulière doit être apportée à l'expression des variables x_i car plusieurs cas doivent être envisagés selon que le résultat du programme linéaire est borné ou non. Si la solution n'est pas bornée alors $x = \sum_j \lambda_j . v_{ij}$ mais la contrainte de convexité n'est pas mise à jour. Dans le cas contraire (la solution est bornée) la contrainte de convexité est mise à jour.

La mise en œuvre Java est un peu délicate dans la mesure ou il faut mémoriser les résultats des sous-problèmes pour calculer la nouvelle fonction objectif du problème maître. Il faut de plus prendre les précautions habituelles liées à la manipulation des nombres réels. Ainsi la partie du pseudo-code :

```
Si  Z^i < σ_i  alors
```

se traduira en Java :

```
if (Z <(x_convexe_1-epsilon))
```

De même, un test d'égalité sera remplacé par une test d'égalité à un epsilon près. Notons que pour des petits exemples, ces précautions sont de peu d'intérêt. Le lecteur constatera toutefois par la suite, qu'elles ont été nécessaires pour un certain nombre d'exemples.

L'idée de la méthode

Rappelons que, si l'on dispose de *k* vecteurs M_1, M_2, …, M_k de IR^n, alors l'ensemble des vecteurs *M* définis par $M = \sum_{i=1}^{k} \lambda_i M_i$ (où les λ_i sont des réels) décrit le sous-espace défini par les vecteurs M_i. Si, de plus, on a, pour tout *i*, $\lambda_i \geq 0$ et $\sum_{i=1}^{k} \lambda_i = 1$, alors cet ensemble de vecteurs est l'enveloppe convexe des vecteurs M_i. Par exemple, si l'on a quatre vecteurs M_1, M_2, M_3, M_4 situés dans un même plan *P* et non alignés, alors l'ensemble des vecteurs *M* définis par $M = \lambda_1 M_1 + \lambda_2 M_2 + \lambda_3 M_3 + \lambda_4 M_4$ (où les λ_i sont des réels) est *P*. Si l'on se restreint aux $\lambda_i \geq 0$ tels que $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$, alors on trouve (en général) le quadrilatère convexe de sommets les vecteurs M_1, M_2, M_3, M_4 et, dans certains cas particuliers, un triangle (dans le cas où l'un des vecteurs est intérieur au triangle défini par les trois autres). Remarquons que, si les vecteurs M_1, M_2, M_3 sont non alignés, alors l'ensemble des vecteurs *M* définis par $M = \lambda_1 M_1 + \lambda_2 M_2 + \lambda_3 M_3$ (où les λ_i sont des réels) est *P*. Si l'on se restreint aux $\lambda_i \geq 0$ tels que $\lambda_1 + \lambda_2 + \lambda_3 = 1$, alors on trouve le triangle de M_1, M_2, M_3. Enfin, si les vecteurs M_1 et M_2 sont distincts, alors l'ensemble des vecteurs *M* définis par $M = \lambda_1 M_1 + \lambda_2 M_2$ (où les λ_i sont des réels) est la droite définie par M_1 et M_2. Si l'on se restreint aux $\lambda_i \geq 0$ tels que $\lambda_1 + \lambda_2 + \lambda_3 = 1$, alors on trouve le segment d'extrémités M_1 et M_2.

Pour simplifier, nous supposerons que les différents polyèdres apparaissant dans la suite sont bornés, ce qui, en pratique, peut toujours être garanti en ajoutant une contrainte destinée à éviter les dépassements de capacité. Considérons donc un problème comme :

$$
\begin{aligned}
&\max\ 7.x_1 + 5.x_2 + 3.x_3 + 7.x_4 + 5.x_5 + 3.x_6 + 2.x_7 \\
&C1 : x_1 + 2x_2 + x_3 + 2x_4 + x_5 + 2x_6 + x_7 \leq 10, \\
&C2 : x_1 + x_2 + x_3 \leq 5, \\
&C3 : 2x_2 + x_3 \leq 8, \\
&C4 : x_4 + 2x_5 + 4x_6 \leq 3, \\
&C5 : x_i \geq 0.
\end{aligned}
\qquad \text{(P)}
$$

Ce problème est dans IR^7. Nous le partageons en deux sous-problèmes :

$$\begin{aligned}&\max\ 7.x_1+5.x_2+3.x_3\\&C'1: x_1+x_2+x_3\le 5,\\&C'2: 2x_2+x_3\le 8,\\&C'3: x_1\ge 0, x_2\ge 0, x_3\ge 0.\end{aligned}$$

et

$$\begin{aligned}&\max\ 7.x_4+5.x_5+3.x_6+2.x_7\\&C''1: x_4+2x_5+4x_6\le 3,\\&C''2: x_4\ge 0, x_5\ge 0, x_6\ge 0, x_7\ge 0.\end{aligned}$$

en éliminant la contrainte couplante (*C*1).

Appelons K le polyèdre défini par les contraintes de départ (*C*1 à *C*5) et prenons deux vecteurs extrémaux P_1 et P_2 du polyèdre défini (dans IR^3, puisque n'interviennent que les 3 premières variables) par les contraintes *C'*1, *C'*2 et *C'*3. Le segment d'extrémités P_1 et P_2 est formé des vecteurs *M* tels que $M=\lambda_1 P_1+\lambda_2 P_2$ avec $\lambda_1\ge 0$, $\lambda_2\ge 0$ et $\lambda_1+\lambda_2=1$. Faisons de même pour le second sous-problème (dans IR^4) à partir des points Q_1 et Q_2. En accolant les 3 coordonnées des vecteurs du segment d'extrémités P_1 et P_2 et les 4 coordonnées des vecteurs du segment d'extrémités Q_1 et Q_2, on suppose que l'on obtient au moins un vecteur du polyèdre *K*. Alors parmi l'ensemble des vecteurs ainsi obtenus (qui décrivent un polyèdre convexe) et appartenant au polyèdre *K*, on recherche celui qui maximise la fonction de coût puis on cherche, en utilisant l'idée de la relaxation lagrangienne, d'autres vecteurs plus prometteurs permettant de mieux optimiser la fonction de coût en décrivant plus complètement les deux polyèdres des sous-problèmes, donc le polyèdre *K*. En complétant ainsi progressivement la description des deux polyèdres des sous-problèmes, on parcourt une partie prometteuse du polyèdre *K* jusqu'à optimiser la fonction de coût sur *K* entier.

4 - Principe de la décomposition de Benders

Description

La méthode est dédiée aux problèmes linéaires mixtes. Elle consiste à résoudre un problème maître ne contenant que des variables réelles et plusieurs sous-problèmes à variables entières [BEN 62].

Le problème maître fournit à chaque itération une borne inférieure de la solution optimale et chaque sous-problème fournit une borne supérieure.

La méthode peut se justifier de la manière suivante en distinguant les variables réelles et les variables entières.

$$\begin{aligned}&\max_{x,y}\ c_1.x+c_2.y\\&C1: A_1.x+A_2.x\le b\\&C2: x\ge 0,\\&C3: y\ge 0, \text{ entiers.}\end{aligned}$$

Pour trouver une borne inférieure, on peut remarquer qu'on peut réécrire le programme précédent, en fixant y (y n'est plus une variable mais une constante). Le programme linéaire précédent peut s'écrire :

$$\begin{aligned}&\max_{x}\ c_1.x+c_2.y\\&C1: A_1.x\le b-A_2.y,\\&C2: x\ge 0.\end{aligned}$$

Notons $V^0(y)$ la solution optimale.

Comme $c_2.y$ est constante, on peut résoudre simplement :

$$\begin{aligned}&\max_{x}\ c_1.x\\&C1: A_1.x\le b-A_2.y,\\&C2: x\ge 0.\end{aligned}$$

Notons (D1) le dual de ce problème.

$$\min_u u(b - A_2 y)$$

(D1) : $C1 : u.A_1 \geq c_1,$ et $W^0(y)$ la solution optimale.

$C2 : u \geq 0,$

De manière évidente, $W^0(y) + c_2 y = V^0(y)$.

Donc puisque y est fixé, $W^0(y) + c_2 y = V^0(y)$ est une borne inférieure de la solution recherchée.

Pour obtenir une borne supérieure, nous remarquons que le programme initial pourrait être résolu en énumérant tous les points *y* correspondants. Il pourrait être réécrit :

$$\max Z$$
$$C1 : Z \leq c_2.y + u_1(b - A_2.x)$$
$$C2 : Z \leq c_2.y + u_2(b - A_2.x)$$
$$C3 : \ldots$$
$$C4 : Z \leq c_2.y + u_K(b - A_K.x)$$
$$C5 : y \geq 0, \text{ entiers.}$$

Comme il est impossible de générer la totalité de ces contraintes, on va les générer successivement (il s'agit donc d'une méthode de génération de colonnes) en espérant obtenir la solution optimale en ajoutant un très faible nombre de ces contraintes. On note P^r le programme linéaire limité à r contraintes.

(P^r) :
$$\max Z$$
$$C1 : Z \leq c_2.y + u_1(b - A_2.x)$$
$$C2 : Z \leq c_2.y + u_2(b - A_2.x)$$
$$C3 : \ldots$$
$$C4 : Z \leq c_2.y + u_r(b - A_r.x)$$
$$C5 : y \geq 0, \text{ entiers.}$$

La résolution de ce problème fournit de manière évidente une borne supérieure au problème initial puisque seules r contraintes figurent sur les K contraintes nécessaires. Remarquons que dans le cas d'un problème de minimisation, le rôle du problème maître et des sous-problème est inversé : le problème maître fournit une borne supérieure du problème de départ. Ce point est mis en évidence dans le deuxième exemple présenté ci-dessous.

De ces deux remarques découlent l'algorithme suivant :

<u>Algorithme de principe :</u>

```
Choisir un vecteur y initial
iter := 0
Stop :=faux
Repeter
   iter := iter + 1
   Contruire le programme linéaire (D1)
   Avec u = b − A2 y et les contraintes du dual.
   Résoudre le problème D1 et notons Z^1 la solution optimale.
```

$LB := Z^1$

Constuire le programme P^{iter} en ajoutant la contrainte

$Z \le C_2.y + u(b - A_2 y)$ au programme P^{iter-1}

Résoudre le problème P^{iter} et notons Z^2 la solution optimale.

$UB := Z^2$

Si (*LB=UB*) **Alors**

Stop := vrai ;

Fin Si

Jusqu'à ce que (stop=vrai).