Chapitre 6

Modélisation en P.L.I.

6.1 Lien entre PL et PLI

(P) problème de PL.

- On restreint les variables à être entières : on a un problème de PLI (ILP en anglais).
- On restreint certaines variables à être entières : on a un problème mixte (MILP en anglais).
- On peut aussi contraindre les variables à être dans $\{0, 1\}$: on a un problème de programmation 0-1.

La question de résoudre efficacement les problemes de PLI se pose donc. On sait qu'on peut le faire de manière efficace pour la PL en théorie (le problème est polynomial) et en pratique (bien que exponentiel dans le pire cas, le simplexe fonctionne bien). Peut-on obtenir les mêmes résultats pour le problème de PLI ?

6.1.1 Approximation de la PLI

Une méthode immédiate consiste à résoudre le problème de PL obtenu en relachant la contrainte $x \in \mathcal{N}$ en $x \in \mathcal{R}$, à résoudre le problème de PL obtenu, puis à prendre comme solution la solution entière la plus proche. Plusieurs problèmes se posent :

- Quelle approximation ? partie entière inférieure ou supérieure, la plus proche ?
  $x = 1.22$ est proche de $\lfloor x \rfloor = 1$ et $\lceil x \rceil = 2$, le plus proche est 1. Mais $x = 1.5$ est aussi proche de 1 que de 2.
  Par contre le dessin suivant montre que ça ne donne pas forcément une solution !

– Quelle est le lien entre solution entières et solution réelles ? Le dessin suivant montre qu'elle peuvent être très éloignées.

Le seul lien exact qu'on peut établir (pour un problème de maximisation) est que si z est la valeur de la fonction objectif optimale du problème de PL, si z est la valeur de la fonction objectif du problème de PLI, alors $z \geq \bar{z}$. Cela ets immédiat : rajouter la contrainte d'intégrité limite le nombre de valeurs admissibles donc le nombre de valeurs possibles de la fonction objectif.

(Pour un problème de minimisation, on aura $z \leq \bar{z}$).

6.1.2 Méthodes ad-hoc

On peut chercher à mettre au point des techniques adaptées aux problèmes particuliers de PLI.

– Programmation dynamique qui est une méthode à base de tabulation pour stocker des résultats intermédiaires qui ne sont donc calculés qu'une fois.
– Méthodes branch and bound (Séparation et Evaluation, en français). Méthodes qui permettent de trouver la solution en énumérant -de manière intelligente- toutes les valeurs entières possibles pour les variables.
– Méthode particulières comme les coupes de Gomory.

6.1.3 Limite théorique

En se reportant au cours de complexité, vous verrez que le problème de programmation linéaire en nombre entiers est un problème NP-complet : ce sont les problèmes les plus difficiles de la classe NP et si on savait les résoudre en temps polynomial, alors on aurait P=NP, ce qui est considéré comme improbable (encore que[1]...). Donc tout algorithme déterministe polynomial qu'on trouvera pour résoudre le problème demandera au moins un temps exponentiel.

En fait le simple fait problème de résoudre un système d'équation diophantiennes (donc une conjonction d'égalités) qui correspond à un cas très particulier de la PLI puisqu'aucune fonction objectif ne doit être optimisée et que tous les coefficients sont entiers est déjà un problème NP-complet.

[1] voir l'article de Delahaye dans pour la science numéro de 2005

Il n'est d'ailleurs pas évident de montrer que le problème de PLI (à coefficients entiers) est dans la classe NP. Pour cela une majoration de la taille des solutions minimales est nécéssaire et il faut également supposer que les entiers sont répresentés en base $b > 1$ (sinon une exponentielle supplémentaire intervient).

6.2 Exemples

6.2.1 Le voyageur de commerce

Un voyageur de commerce doit faire sa tournée en visitant n villes $x_1, \ldots, x_n$ en ne visitant chaque ville x_i, $(i \neq 1)$ qu'une seule fois en partant de x_1 puis en y retournant. La distance entre la ville x_i et la ville x_j est donnée par $c_{i,j} > 0$ $(i \neq j)$et la question est de minimiser la distance parcourue dans la tournée. L'ensemble des villes se modélise par un graphe complet étiquetté (toute ville est accessible d'une autre vile). Ce problème est connu comme celui du voyageur de commerce (TSP en anglais).

Remarque : on peut avoir $c_{i,j} = c_{j,i}$ pour tout i, j (problème symétrique) ou bien $c_{i,j} \neq c_{j,i}$ pour certains i, j. Nous traiterons ici le deuxième cas qui est le plus général.

- Choix des variables : $x_{i,j}$ (n^2 variables) qui vaut 1 si la tournée emprunte l'arc i à j.
- Modéliser le chemin :
  - je n'arrive qu'une seule fois à la ville i : un seul arc entrant sur i est emprunté $\Sigma_{j=1}^{j=n} c_{j,i} x_{j,i} = 1$
  - une seule ville est visitée en partant de i : un seul arc sortant sur i est emprunté $\Sigma_{j=1}^{j=n} c_{i,j} x_{i,j} = 1$
- La distance à minimiser : $z = \Sigma_{i \in \{1,..,n\}} \Sigma_{j \neq i, j \in \{1,..,n\}} x_{i,j} c_{i,j}$

On a donc de l'ordre de $O(n^2)$ variables et équations.

Cette modélisation semble correcte, malheureusement elle est incomplète. Il peut y avoir des sous-cycles comme dans l'exemple suivant :

Comment éliminer ces solutions parasites ?

Ajout de contraintes On rajoute à la modélisation, la contrainte qui exprime que pour tout sous-ensemble S non vide ou non égal à$\{1, ..., n\}$, il y a au moins un chemin entre S et son complémentaire $\bar{S}$:

$$\Sigma_{i \in S, j \in \bar{S}} x_{i,j} \geq 1$$

Cela donne un nombre de contraintes en $O(2^n)$, ce qui est très mauvais car on introduit un nombre exponentiel de contraintes dont la résolution est couteuse.

Une approche plus pragmatique (utilisée dans la réalité) consiste à utiliser la première modélisation. Si on a une solution sans cycle on a gagné (car la modélisation autorise plus de solutions que dans la réalité et donc si la solution correspond à une vraie tournée alors c'est bien celui cherché). Sinon on a des cycles indépendants on "casse" le cycle en rajoutant une contrainte qui exprime qu'il y a au moins un arc emprunté entre deux sous-cycles. Cela revient à ne rajouter les contraintes

$$\Sigma_{i \in S, j \in \bar{S}} x_{i,j} \geq 1$$

avec S un ensemble de sommets qui correspond à un cycle que lorsque nécéssaire. Evidemment rien ne garantit que le processus ne termine avant d'avoir rajouté un nombre de contrainte exponentiel.

Ajout de variables réelles Un autre rapproche consiste à rajouter pour chaque $i = 2, ..., n$ une variable $u_i \in \mathcal{R}$ (noter que $i = 1$ ne donne aucune variable). De plus on rajoute les contraintes

$$u_i - u_j + nx_{i,j} \leq n - 1$$

pour $i \neq j \in \{2, \ldots, n\}$.

On montre qu'alors le système obtenu est équivalent à résoudre le problème du TSP.

Fait 1. Si la résolution des contraintes donne une solution qui contient un cycle, alors celui-ci contient nécéssairement la ville 1 (un seul cycle est possible et c'est donc la solution voulue).

On suppose qu'il existe un cycle $i_1, \ldots, i_p$ avec $p \geq 2$ qui ne contient pas la ville 1. Tous les $x_{i_j,i_{j+1}}$ ainsi que x_{i_p,i_1} valent 1 ce qui permet d'écrire

$$\begin{array}{rcl} u_{i_1} - u_{i_2} + n & \leq & n-1 \\ u_{i_2} - u_{i_3} + n & \leq & n-1 \\ & \cdots & \\ u_{i_p} - u_{i_1} + n & \leq & n-1 \\ \hline 0 & \leq & -p \end{array}$$

absurde.

Cela permet d'être sur qu'une solution du système de contrainte donne un chemin qui correspond bien à une tournée.

Fait 2. Une solution $i_1 = 1, \ldots, i_p = n, i_1$ pour le TSP donne une solution pour le système de contraintes.

On a donc $x_{i,j} = 1$ ssi le chemin contient un arc de i à j. Le chemin donne un ordre de visite des villes et on donne à u_i ($i \geq 2$) le numéro d'ordre correspondant. Si i_1 est la première ville visitée alors on donne 1 comme valeur à u_i, etc... Alors toutes les contraintes supplémentaires sont vérifiées :

- i, j telles que $x_{i,j} = 0$. Alors on a $u_i, u_j \in \{1, \ldots, n-1\}$ et donc la différence $u_i - u_j$ est donc inférieure à $n - 1$, c.a.d.

$$u_i - u_j + nx_{i,j} \leq n - 1$$

- i, j telles que $x_{i,j} = 1$. Alors la ville i est la $k$eme ville visitée et la ville j est la $k + 1$eme ville visitée et donc $k - (k + 1) + n = n - 1 \leq n - 1$, c.a.d.

$$u_i - u_j + nx_{i,j} \leq n - 1$$

En combinant les deux faits précédents on voit qu'une solution des contraintes donne bien une tournée et que toute tournée correspond à une solution des contraintes. Par conséquent une solution minimale résoud bien le problème du TSP.

6.2.2 Sac à dos

Un voyageur peut rapporter un certain nombre d'objets $n = 4$ avec lui dans son sac à dos, sachant qu'il peut les revendre avec un bénéfice qu'il connait.

La contenance de son sac à dos est $V = 14l$, et chaque objet i a un volume $v_i > 0$ repectivement de $v_1 = 5, v_2 = 7, v_3 = 4, v_4 = 3$.

Le bénéfice pour chaque objet est $c_1 = 8, c_2 = 11, c_3 = 6, c_4 = 4$.

La question est de maximiser le bénéfice.

Une modélisation est de prendre une variable x_i qui vaut 0 si on ne prend pas l'objet i et qui vaut 1 si on le prend.

D'où :

$$\begin{array}{rcl} Max\ z & = & \Sigma_{i=1}^{i=n} c_i x_i \\ & & \Sigma_{i=1}^{i=n} v_i x_i \leq V \\ & & x_i \in \{0,1\} \end{array}$$

qui donne

$$\begin{array}{rl} Max\ z & = 8x_1 + 11x_2 + 6x_3 + 4x_4 \\ & 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \\ & x_i \in \{0,1\} \end{array}$$

C'est donc un problème de programmation 0-1