Outils de Recherche Opérationnelle en Génie - MTH 8414

Programmation en nombres entiers

Astuce de modélisation

Louis-Martin Rousseau

Office: A520.21 Tel.: #4569
Louis-Martin.Rousseau@polymtl.ca

POLYTECHNIQUE MONTRÉAL

Trucs et astuces de modélisation

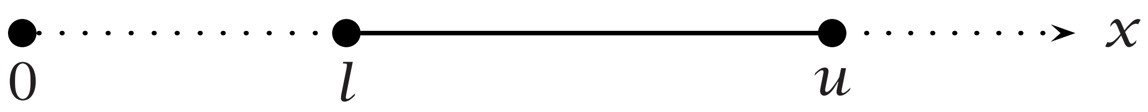
Comment modéliser les cas où l’on est en présence de:

- variables ont des domaines discontinus;
- certaines ressources qui ont des coûts fixes;
- disjonctions de contraintes;
- contraintes conditionnelles
- des produits de variables
- de SOS et des fonctions linéaires par morceaux

POLYTECHNIQUE MONTRÉAL

Variables avec domaines discontinues

- Que faire avec le cas où soit $x = 0$ OU $l <= x <= u$?



- On peut considérer ceci comme deux contraintes, mais elles ne peuvent être vraies toutes les deux à la fois…

  - Pouvez-vous trouver des exemples d’applications ?

  - Comment modéliser ceci avec un PLNE ?

Variables avec domaines discontinues

- On utilisera une variable indicatrice :

$$y = \begin{cases} 0 & \text{for } x = 0 \\ 1 & \text{for } l \leq x \leq u \end{cases}$$

- Qu'on liera avec la variable originale par les contraintes suivantes :

$$\begin{aligned} & x \leq uy \\ & x \geq ly \\ & y \quad \text{binary} \end{aligned}$$

- *Y = 0* implique donc *x = 0* et *y = 1* implique que $l \leq x \leq u$.

Les coûts fixes

- Soit le problème suivant:

Minimize: $$C(x)$$

Subject to:

$$a_i x + \sum_{j \in J} a_{ij} w_j \gtrless b_i \qquad \forall i \in I$$

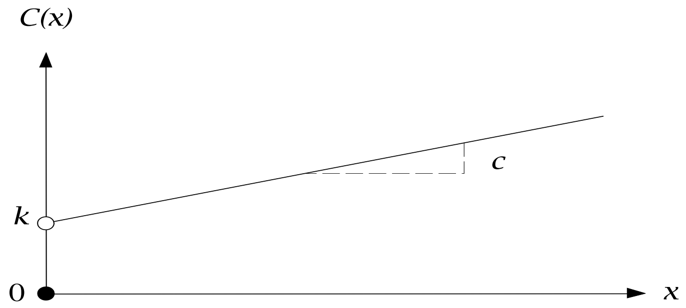
$$x \geq 0$$

$$w_j \geq 0 \qquad \forall j \in J$$

Where:

$$C(x) = \begin{cases} 0 & \text{for } x = 0 \\ k + cx & \text{for } x > 0 \end{cases}$$

- La fonction de coût n'est ni linéaire ni continue...
- À quelle application pensez-vous ?
- Comment résoudre ce problème ?



Les coûts fixes

- Si on connaît une borne u suffisamment grande pour x et qu'on introduit une variable indicatrice y

$$y = \begin{cases} 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0 \end{cases}$$

- On relie x et y par $x \leq yu$

- L'objectif devient donc: $$C^*(x, y) = ky + cx$$

- Et le problème devient:

Minimize: $$ky + cx$$

Subject to:

$$a_i x + \sum_{j \in J} a_{ij} w_j \gtrless b_i \qquad \forall i \in I$$

$$x \leq uy$$

$$x \geq 0$$

$$w_j \geq 0 \qquad \forall j \in J$$

$$y \quad \text{binary}$$

Une disjonction de contrainte

- Soit le problème suivant:

Minimize:

$$\sum_{j \in J} c_j x_j$$

Subject to:

$$\sum_{j \in J} a_{1j} x_j \leq b_1 \qquad (1)$$

$$\sum_{j \in J} a_{2j} x_j \leq b_2 \qquad (2)$$

$$x_j \geq 0 \qquad \forall j \in J$$

- Où soit (1) ou (2) doit être respectée
- Des applications ?
- Comment faire ?

POLYTECHNIQUE MONTRÉAL

Une disjonction de contraintes

Encore une fois on introduira une variable supplémentaire *y* ainsi que deux grands nombres (M_1 et M_2).

En modifiant (1) et (2) de la manière suivante:

$$(1) \quad \sum_{j \in J} a_{1j} x_j \leq b_1 + M_1 y$$

$$(2) \quad \sum_{j \in J} a_{2j} x_j \leq b_2 + M_2 (1 - y)$$

On s'assure qu'une des deux contraintes devra être satisfaite.

Minimize: $\sum_{j \in J} c_j x_j$

Subject to:

$$\sum_{j \in J} a_{1j} x_j \leq b_1 + M_1 y$$

$$\sum_{j \in J} a_{2j} x_j \leq b_2 + M_2 (1 - y)$$

$$x_j \geq 0 \qquad \forall j \in J$$

$$y \quad \text{binary}$$

Contraintes conditionnelles

Une variante de ce problème survient lorsque certaines contraintes sont conditionnelles:

$$\textit{If} \quad \textit{(1)} \quad \Big(\sum_{j \in J} a_{1j} x_j \leq b_1\Big) \quad \textit{is satisfied,}$$

$$\textit{then} \quad \textit{(2)} \quad \Big(\sum_{j \in J} a_{2j} x_j \leq b_2\Big) \quad \textit{must also be satisfied.}$$

- Donnez des exemples d'application ?
- Comment traiter ce cas ?

POLYTECHNIQUE MONTRÉAL

Contraintes conditionnelles

Pour adresser ce cas, nous devons nous tourner vers la logique

L'équation logique qui nous intéresse est (A implique B)

Cette équation est équivalente à (non-A OU B)

On a donc une disjonction de contraintes, qu'on peut traiter comme précédemment…

Devient: $$\textit{If} \quad \Big(\sum_{j \in J} a_{1j}x_j \leq b_1\Big) \quad \textit{holds, then} \quad \Big(\sum_{j \in J} a_{2j}x_j \leq b_2\Big) \quad \textit{must hold,}$$

Sauf qu'ici on a un signe > qu'on ne peut traiter en PL…

$$\Big(\sum_{j \in J} a_{1j}x_j > b_1\Big) \quad \textit{or} \quad \Big(\sum_{j \in J} a_{2j}x_j \leq b_2\Big) \quad \textit{must hold.}$$

Contraintes conditionnelles

On introduira une petite valeur *epsilon* $\sum_{j \in J} a_{1j} x_j \geq b_1 + \epsilon$

Pour obtenir: $$\sum_{j \in J} a_{1j} x_j \geq b_1 + \epsilon, \quad or \quad \sum_{j \in J} a_{2j} x_j \leq b_2 \quad must\ hold.$$

Qui peut être réécrit comme:

$$\sum_{j \in J} a_{1j} x_j \geq b_1 + \epsilon - Ly$$
$$\sum_{j \in J} a_{2j} x_j \leq b_2 + M(1 - y)$$

Éliminer les produits de variables

Que faire des problèmes où des termes contiennent le produit de deux variables booléennes x_1x_2

Éliminer les produits de variables

Que faire des problèmes où des termes contiennent le produit de deux variables booléennes x_1x_2

On peut faire disparaître ce produit en introduisant une nouvelle variable booléenne y qui doit être égale au produit x_1x_2.

Pour ce faire il faut ajouter les contraintes suivantes:

$$
\begin{aligned}
&y \leq x_1 \\
&y \leq x_2 \\
&y \geq x_1 + x_2 - 1 \\
&y \quad \text{binary}
\end{aligned}
$$

POLYTECHNIQUE MONTRÉAL

Éliminer les produits de variables

Que faire maintenant si on doit traiter un produit x_1x_2 où x_1 est une variable binaire et x_2 est une variable continue tel que $0 <= x_2 <= u$?

On introduit une variable continue y définie comme $y = x_1x_2$ en ajoutant les contraintes ci-dessous imposant le comportement:

$$y \leq u x_1$$
$$y \leq x_2$$
$$y \geq x_2 - u(1 - x_1)$$
$$y \geq 0$$

x_1	x_2	x_1x_2	**constraints**	**imply**
0	$w : 0 \leq w \leq u$	0	$y \leq 0$ $y \leq w$ $y \geq w - u$ $y \geq 0$	$y = 0$
1	$w : 0 \leq w \leq u$	w	$y \leq u$ $y \leq w$ $y \geq w$ $y \geq 0$	$y = w$

POLYTECHNIQUE MONTRÉAL

Les SOS (Special Ordered Sets)

Considérer le cas particulier suivant:

- Votre modèle comporte une série de décision oui/non ordonnée.
- Seulement une décision « oui » est possible
- Entre deux décisions « oui » on préfèrera toujours celle qui est la première dans la série.

Pour ce cela, on dispose généralement d'une série de variables booléennes y_i telles que $\sum_i y_i \leq 1$

On peut généraliser ce cas à :

- Aux variables générales x_i tel que $0 <= x_i <= u$ et sujet à $\sum_i a_i x_i \leq b$

On peut aussi considérer le cas où deux décisions « oui » sont permises, mais elles doivent être consécutives.

Les solveurs ont des objets de modélisation SOS1 et SOS2 qui implémentent ces conditions de manières plus efficaces lors du branch and bound.

Fonctions linéaires par morceaux

Soit le problème suivant:

Minimize: $$\sum_{j \in J} f_j(x_j)$$

Subject to:

$$\sum_{j \in J} a_{ij} x_j \gtrless b_i \qquad \forall i \in I$$

$$x_j \geq 0 \qquad \forall j \in J$$

Ici on remarque que l'objective, bien que non linéaire, est séparable.

C'est-à-dire que l'objectif est une somme de fonctions définies sur une variable à la fois.

Séparable

$$x_1^2 + 1/x_2 - 2x_3 = f_1(x_1) + f_2(x_2) + f_3(x_3)$$

$$x_1^2 + 5x_1 - x_2 = g_1(x_1) + g_2(x_2)$$

Non séparable

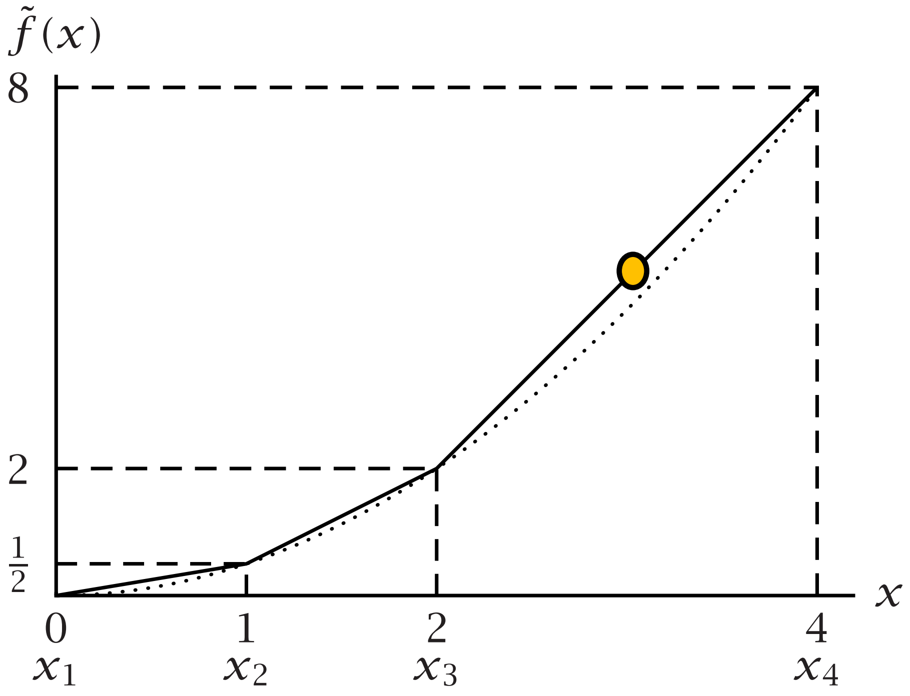
$$x_1 x_2 + 3x_2 + x_2^2 = f_1(x_1, x_2) + f_2(x_2)$$

$$1/(x_1 + x_2) + x_3 = g_1(x_1, x_2) + g_2(x_3)$$

POLYTECHNIQUE MONTRÉAL

Fonctions linéaires par morceaux

Soit la fonction $f(x) = \frac{1}{2}x^2$



Soit x_1, x_2, x_3, x_4 des points de « cassure » (breakpoints) auquel on évalue la fonction f(x) (soit 0,1,2,4)

On approxime donc linéairement tout point situé entre deux cassures, par exemple f(3) = ½*f(2) + ½*f(4) = ½*1 + ½*8 = 5.

POLYTECHNIQUE MONTRÉAL

Fonctions linéaires par morceaux

Une des manières de traiter ces fonctions est d'utiliser la λ-formulation.

Soit λ_1, λ_2, λ_3, λ_4, 4 poids non négatifs dont la somme = 1, alors la fonction linéaire par morceaux précédente peut-être exprimée par:

$$\begin{aligned}
\lambda_1 f(x_1) + \lambda_2 f(x_2) + \lambda_3 f(x_3) + \lambda_4 f(x_4) &= \tilde{f}(x) \\
\lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \lambda_4 x_4 &= x \\
\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 &= 1
\end{aligned}$$

Comme au plus deux λ peuvent être non négatifs, et que ceux-ci doivent en plus être consécutifs, on peut ajouter une contrainte SOS2(λ).

La majorité des solveurs ont un objet « Piecewise Linear » que vous pouvez utiliser directement.