Sujet 8: Programmation linéaire en nombres entiers

MHT 423:
Modélisation et optimisation

Andrew J. Miller
Dernière mise à jour: March 15, 2010

Dans ce sujet...

1. Introduction à programmation linéaire en nombres entiers mixtes
2. Modélisation des problèmes en nombres entiers mixtes
3. Méthodes de resolution pour MIP

Formulations générales

Rappelons la formulation d'un programme linéaire:

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j \\ \text{s.à.} \quad & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \forall i \\ & x_j \geq 0, j = 1, ..., n \end{aligned}$$

La formulation d'un programme en nombres entiers la rassemble; mais maintenant on a aussi des variables qui doivent prendre les valeurs entiers.

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j + \sum_{k=1}^{p} f_k y_k \\ \text{s.à.} \quad & \sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{p} d_{ik} y_k \leq b_i, \forall i \\ & x_j \geq 0, j = 1, ..., n; \\ & y_k \in \mathbb{Z}_+, k = 1, ..., p \end{aligned}$$

Ici, $\mathbb{Z}$ veut dire l'ensemble des nombres entiers, et $\mathbb{Z}_+$ veut dire l'ensemble des nombres entiers non-negatifs.

Relaxations

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} c_j x_j + \sum_{k=1}^{p} f_k y_k \\ \text{s.à.} \quad & \sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{p} d_{ik} y_k \leq b_i, \forall i \\ & x_j \geq 0, j = 1, ..., n; \\ & y_k \in \mathbb{Z}_+, k = 1, ..., p \end{aligned}$$

Notez bien que si on remplace les derniers restrictions par $y_k \geq 0, k = 1, ..., p$, on aura *un programme linéaire* qui sera **une relaxation** du problème original.
Ce fait est très important. Ça implique qu'on peut résoudre la relaxation par des méthodes qu'on a déjà considérés.

Exemple numérique (I)

Problème original:

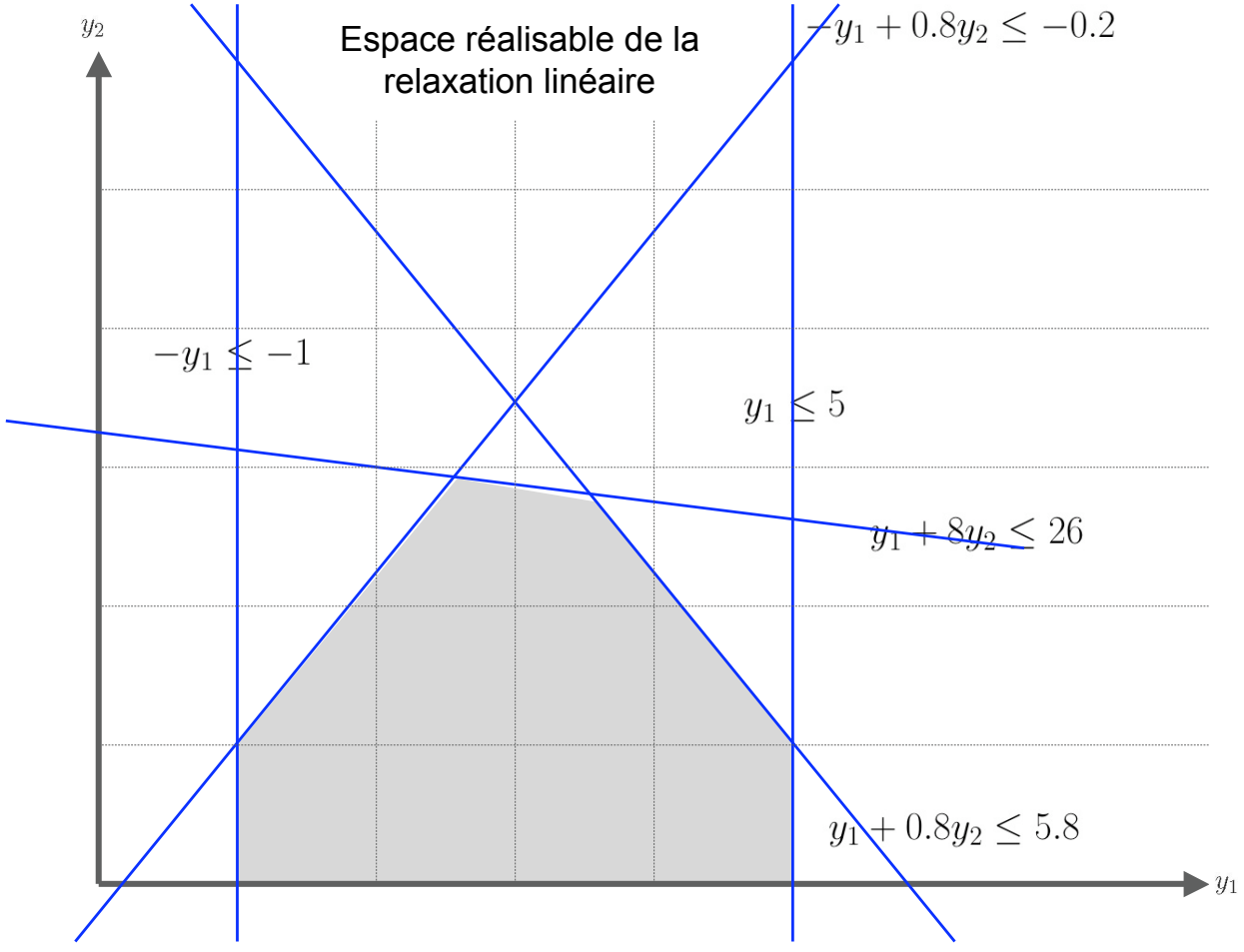
$$
\begin{aligned}
\max \quad & y_1 + 2y_2 \\
\text{s.à} \quad & y_1 + 8y_2 \leq 26 \\
& -y_1 + 0.8y_2 \leq 0.2 \\
& y_1 + 0.8y_2 \leq 5.8 \\
& y_1 \leq 5 \\
& -y_1 \leq -1 \\
& y_1, y_2 \in \mathbb{Z}_+
\end{aligned}
$$

Relaxation du problème:

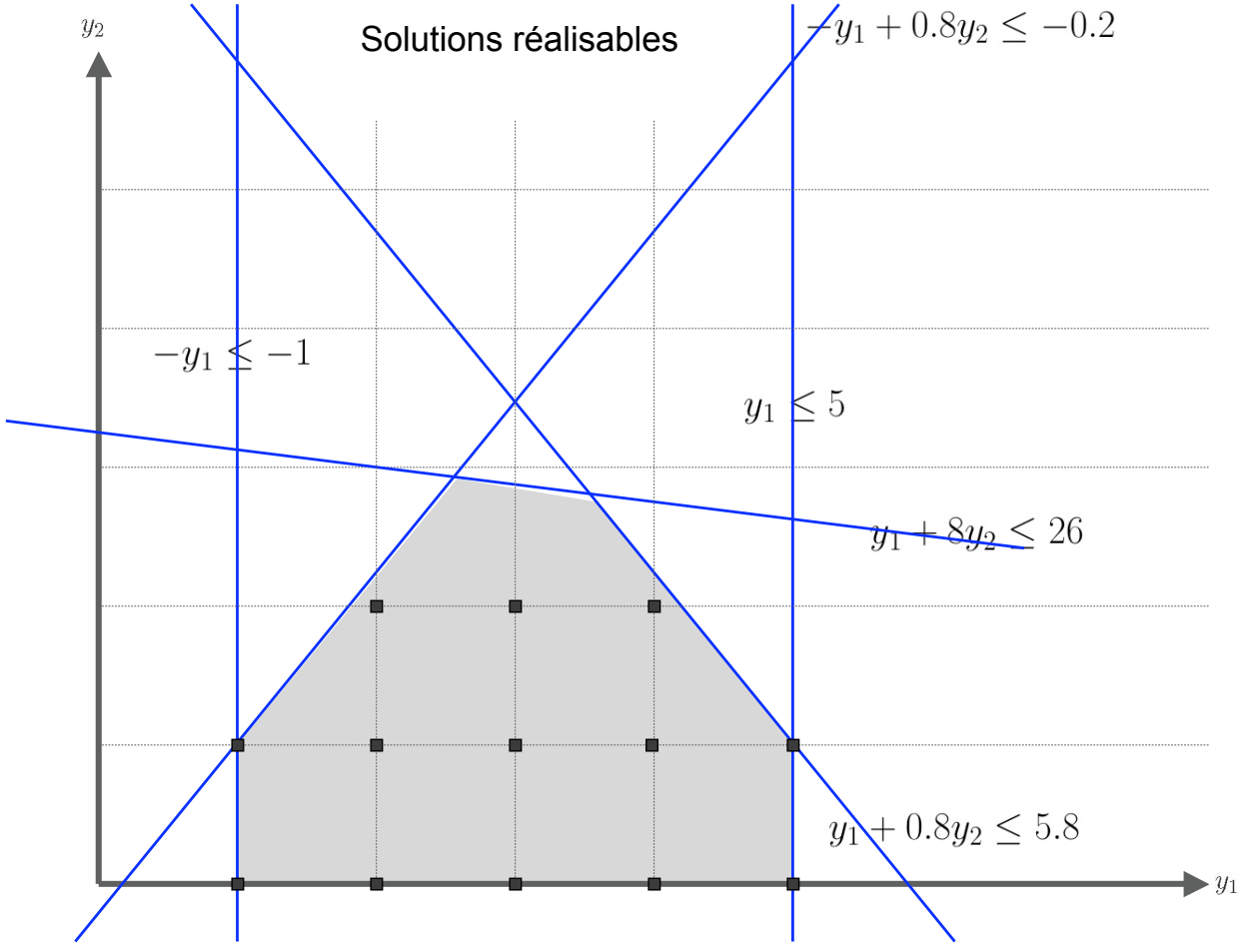
$$
\begin{aligned}
\max \quad & y_1 + 2y_2 \\
\text{s.à} \quad & y_1 + 8y_2 \leq 26 \\
& -y_1 + 0.8y_2 \leq 0.2 \\
& y_1 + 0.8y_2 \leq 5.8 \\
& y_1 \leq 5 \\
& -y_1 \leq -1 \\
& y_1, y_2 \geq 0
\end{aligned}
$$

Ici, $n = 0$ et $p = 2$.

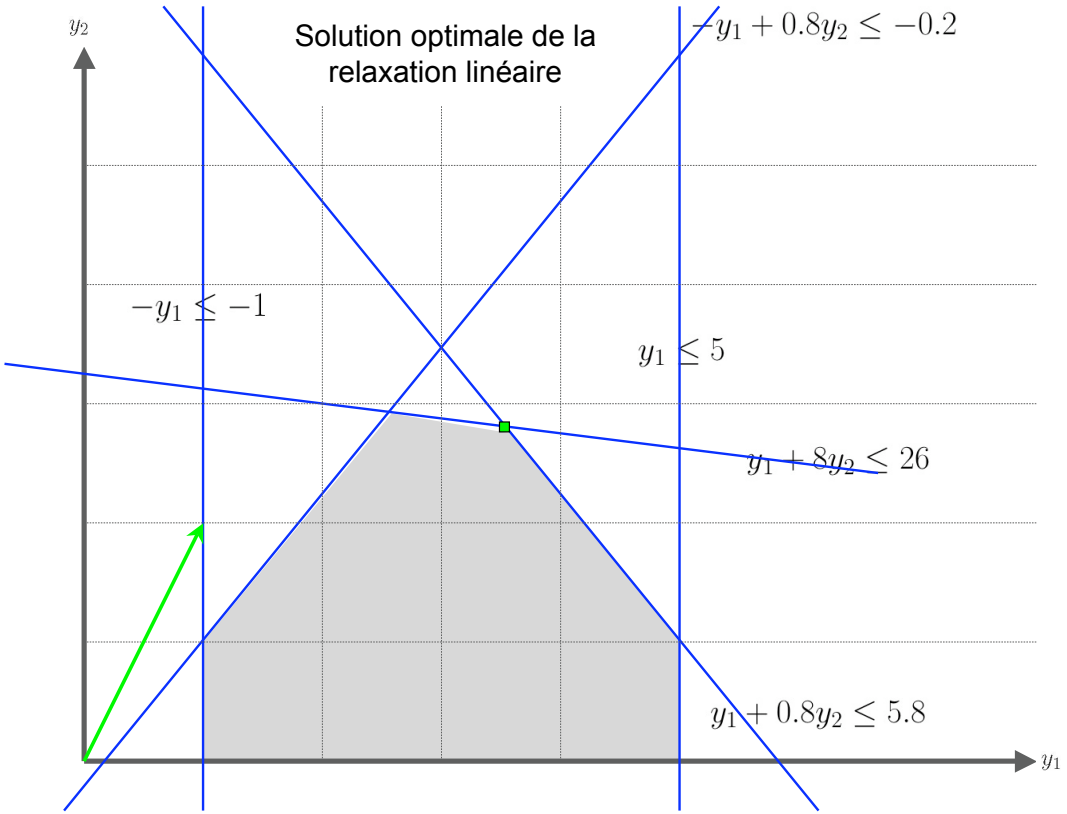
Exemple numérique (II)



Exemple numérique (III)

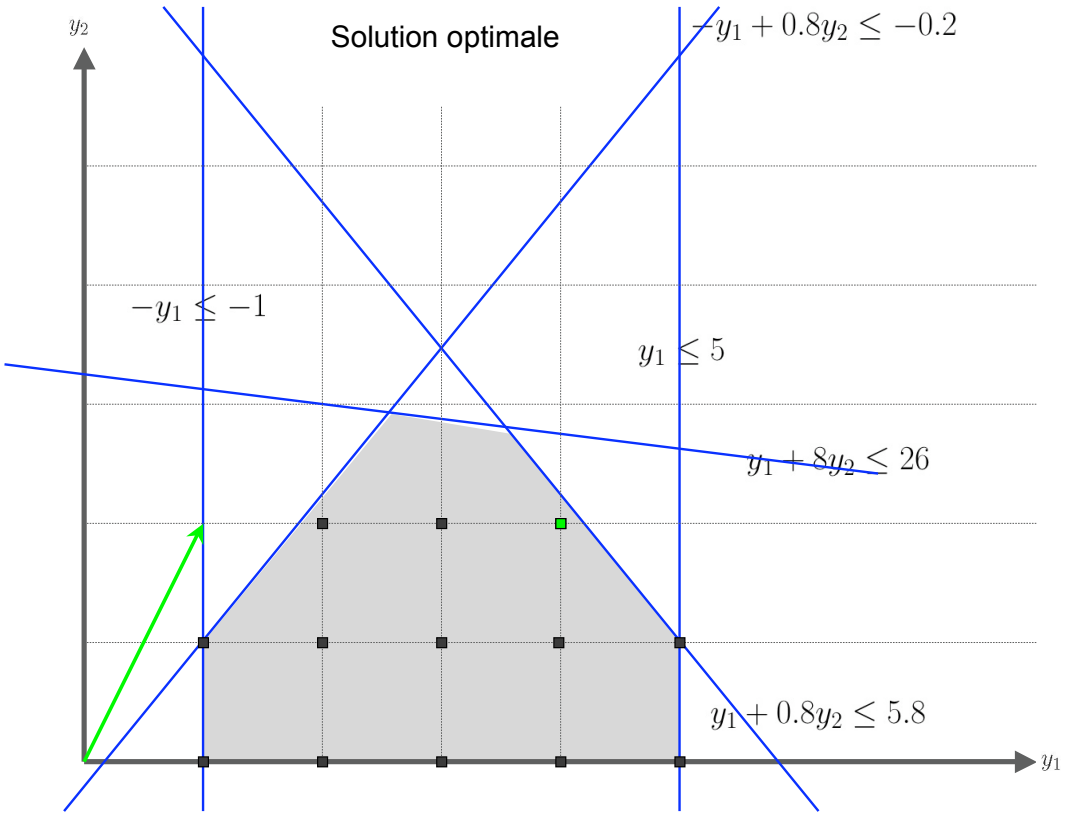


Exemple numérique (IV)



Notez bien la solution optimale de la relaxation: $y_1 = \frac{32}{9} = 3\frac{5}{9}$, $y_2 = \frac{101}{36} = 2\frac{29}{36}$, avec valeur objective de $\frac{55}{6} = 9\frac{1}{6}$.

Exemple numérique (V)



Notez bien la solution optimale du problème original : $y_1 = 1$, $y_2 = 2$, avec valeur objective de 8.

Faits importants illustrés par l'exemple

1. La valeur objective optimale de la relaxation définit *toujours* **une borne supérieure** sur la valeur objective optimale du problème original.

   $$\text{Ex: } 9\frac{1}{6} \geq 8$$

   Encore une fois: **ce fait est très important!**

2. Souvent, même avec la solution optimale de la relaxation, on ne peut pas avoir des autres informations importantes sur le problème par des opérations simples d'arrondi.

   Ex: Bien que la valeur objective optimale de la relaxation est $9\frac{1}{6}$, la valeur objective optimale du problème n'est pas 9.

   Ex: Bien que la solution optimale de la relaxation est $y_1 = 3\frac{5}{9}, y_2 = 2\frac{29}{36}$, la solution optimale du problème n'est pas $y_1 = 4, y_2 = 3$.

Programmation en nombres entiers mixtes et relaxations

Le deuxième fait veut dire que, malheureusement, la programmation en nombres entiers est plus difficil (même beaucoup plus difficil) que la programmation linéaire.

Le premier veut dire qu'on peut néanmoins définir des méthodes de résolution pour les programmes en nombres entiers qui sont basés sur la résolution successive des relaxations linéaires.
Nous verrons les détails un peu plus tard.

Exemple: coûts fixes

Nous avons vu des côuts unitaires. Ces coûts sont proportionnels au niveau de l'activité à laquelle ils s'appliquent.

Par contre, les *côuts fixes* sont indépendant du niveau de l'activités. Ils ne sont pas proportionels.

Soit on fait l'activité, et on pait tout le coût fixe.

Soit on fait pas l'activité, et on ne le pait pas.

Monet: coûts fixes (I)

Rappelons le modèle Monet:

$$
\begin{aligned}
\min \quad & 6x_1 + 2x_2 + 4x_3 + 3x_4 \\
\text{s.à.} \quad & 2x_1 + x_2 + 3x_3 + 2x_4 \leq 4000 \\
& 4x_1 + 2x_2 + x_3 + 2x_4 \leq 6000 \\
& 6x_1 + 2x_2 + x_3 + 2x_4 \leq 10000 \\
& x_1 \leq 1000; x_2 \leq 2000; x_3 \leq 500; x_4 \leq 1000 \\
& x_1, x_2, x_3, x_4 \geq 0
\end{aligned}
$$

Imaginons maintenant que chaque fois qu'on commence la production d'un espèce de cadre-photo, il y a un coût fixe à payer, associé à ce cadre-photo.
Le coût fixe f_i, $i = 1, ..., 4$, est donné par la table suivant

c_1	c_2	c_3	c_4
3100	1200	1000	1800

Monet: coûts fixes (II)

$$
\begin{aligned}
\min \quad & 6x_1 + 2x_2 + 4x_3 + 3x_4 - 3100y_1 - 1200y_2 - 1000y_3 - 1800y_4 \\
\text{s.à.} \quad & 2x_1 + x_2 + 3x_3 + 2x_4 \leq 4000 \\
& 4x_1 + 2x_2 + x_3 + 2x_4 \leq 6000 \\
& 6x_1 + 2x_2 + x_3 + 2x_4 \leq 10000 \\
& x_1 \leq 1000y_1; x_2 \leq 2000y_2; x_3 \leq 500y_3; x_4 \leq 1000y_4 \\
& x_1, x_2, x_3, x_4 \geq 0; y_1, y_2, y_3, y_4 \in \{0, 1\}
\end{aligned}
$$

Il faut remarquer comment on à intégré les nouvelles variables y dans la formulation.
Cette manière assure, pour $i = 1, ..., 4$, que si $x_i > 0$ alors $y_i = 1$.
Elle assure également que le coût fixe pour un bien sera payé si et seulement si la quantité de production de ce bien est strictement positive.

Exemple: décisions binaires

Souvent on utilise les variables binaires pour répresenter les décisions qui sont oui/non (mathématiquement, 1/0).

La variable sera 0 si on ne prend pas la décision. Elle sera 1 si on le prend.

Exemples:

- investir dans un projet ou non
- allouer une certaine tâche à une certaine ou non

Décisions binaires: localisation des dépôts

- I sites pour dépôts éventuels: $1, ..., i, ..., I$
  - Soit f_i le coût fixe d'avoir un dépôt à i.
- J clients $1, ..., j, ..., J$
  - Soit c_{ij} le coût de satisfaire la demande du client j du dépôt i
- Pour simplifier les modèles, nous ne considérons pas les capacités.

Formulation

$$\begin{aligned} \min \quad & \sum_i f_i y_i + \sum_{i,j} c_{ij} x_{ij} \\ \text{soumis à} \quad & \sum_i x_{ij} = 1, \forall j \\ & x_{ij} \leq y_i, \forall i, j \\ & y_j \in 0, 1, \forall j \\ & x_{ij} \in \{0, 1\}, \forall i, j \end{aligned}$$

Le problème "p-Median"

Maintenant, considerer le même problème, mais avec la modification qu'on ne considère plus les coûts fixes associès avec la construction des dépôts. A leur place on imposera la constrainte qu'il faut se limiter à ne construire que p dépôts.

$$\begin{aligned} \min \quad & \sum_{i,j} c_{ij} x_{ij} \\ \text{soumis à} \quad & \sum_{i} x_{ij} = 1, \forall j \\ & x_{ij} \leq y_i, \forall i, j \\ & \sum_{i} y_i = p \\ & y_j \in 0, 1, \forall j \\ & x_{ij} \in \{0, 1\}, \forall i, j \end{aligned}$$

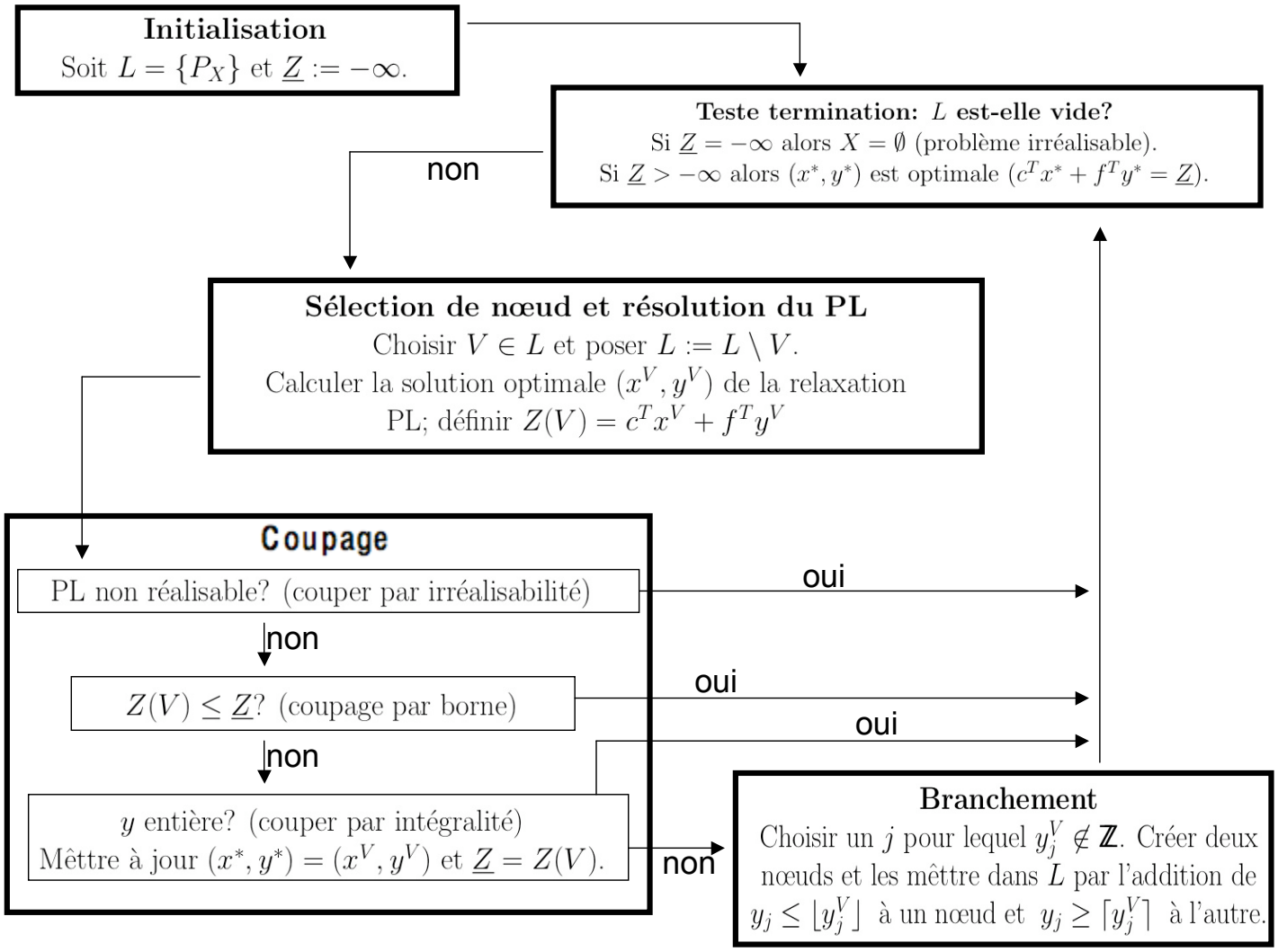
Branch-and-bound

On crée un arbre de recherche et d'énumeration. A chaque noeud, on résoud un programme linéaire.

Ce qu'on fait dépend sur les valeurs des variables qui doivent prendres des valeurs entières.
Deux choix principales:

1. Couper le nœud (a cause des bornes; borne = "bound"). Il y a trois raisons pour lesquelles on peut couper un nœud:
   - La relaxation linéaire n'est pas de solution réalisable.
   - La valeur objective de la relaxation linéaire n'est pas meilleurs que la valeur objective de la meilleure solution entière déjà trouvée.
   - La solution optimale de la relaxation linéaire est déjà entière.
2. Diviser le nœud ("branch"). Si on ne peut pas couper le nœud, il faut le diviser en deux (en effet, il faut créer deux programmes linéaires nouveaux).

Branch-and-bound: organigramme



L'organigramme

Définitions utiles pour le comprendre:

- L est la liste des nœuds (rappeler que chaque nœud correspond à un programme linéaire).
- X est l'espace réalisable du problème.
- P_X, qui définit le premier nœud, est la relaxation linéaire du probème.
- $\underline{Z}$, la borne globale inférieure, est la valeur de la meilleur solution entière trouvée.
- $Z(V)$ est la valeur objective de la solution optimale du programme linéaire défini par le nœud V.

L'organigramme et exemples

Qu'est-ce qui va arriver si on l'applique à l'exemple? Ça peut se faire à la main, et ça vaut vraiment la peine de le faire.

On va commencer (mais pas finir) maintenant et ici.

Autres exemples: localisation des dépôts. Normalement ces problèmes sont trop grands pour appliquer l'organigramme à la main. Mais c'est bien la version basique de ce que font les ordinateurs pendant la resolution.

Exemple numérique (rappel)

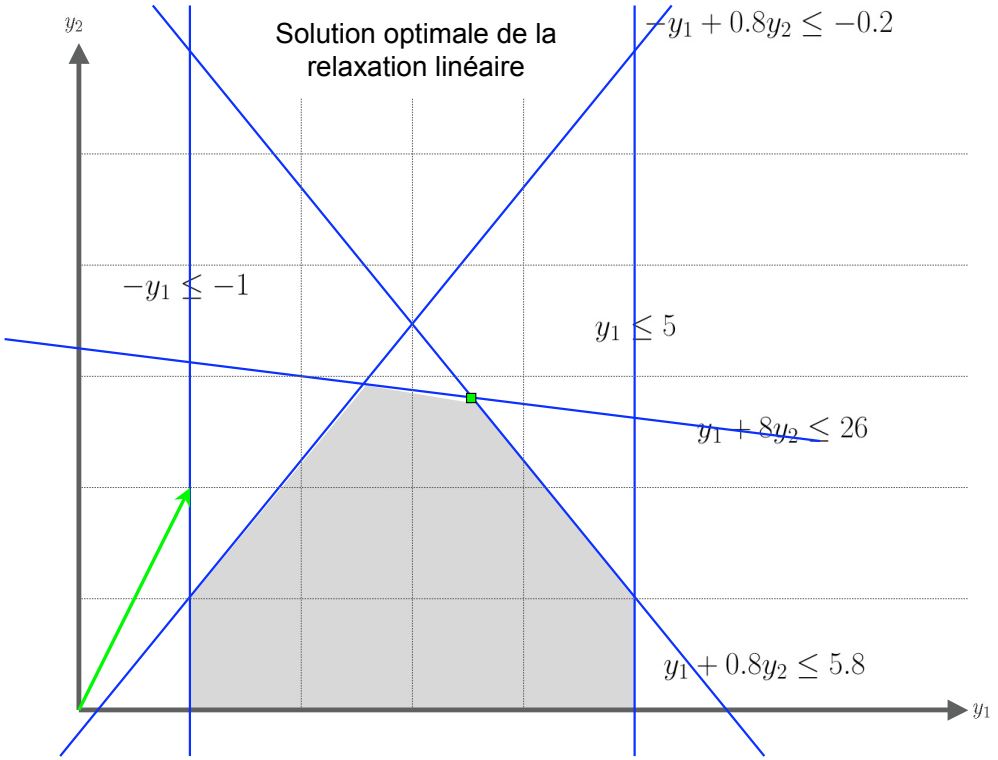
Problème original:

$$
\begin{aligned}
\max \quad & y_1 + 2y_2 \\
\text{s.à} \quad & y_1 + 8y_2 \leq 26 \\
& -y_1 + 0.8y_2 \leq 0.2 \\
& y_1 + 0.8y_2 \leq 5.8 \\
& y_1 \leq 5 \\
& -y_1 \leq -1 \\
& y_1, y_2 \in \mathbb{Z}_+
\end{aligned}
$$

Relaxation initiale:

$$
\begin{aligned}
\max \quad & y_1 + 2y_2 \\
\text{s.à} \quad & y_1 + 8y_2 \leq 26 \\
& -y_1 + 0.8y_2 \leq 0.2 \\
& y_1 + 0.8y_2 \leq 5.8 \\
& y_1 \leq 5 \\
& -y_1 \leq -1 \\
& y_1, y_2 \geq 0
\end{aligned}
$$

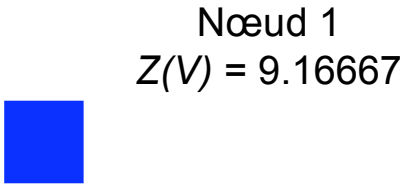
Nœud 1



Rappelez que la solution optimale de la relaxation est $y_1 = \frac{32}{9} = 3\frac{5}{9}$, $y_2 = \frac{101}{36} = 2\frac{29}{36}$, avec valeur objective de $\frac{55}{6} = 9\frac{1}{6}$.

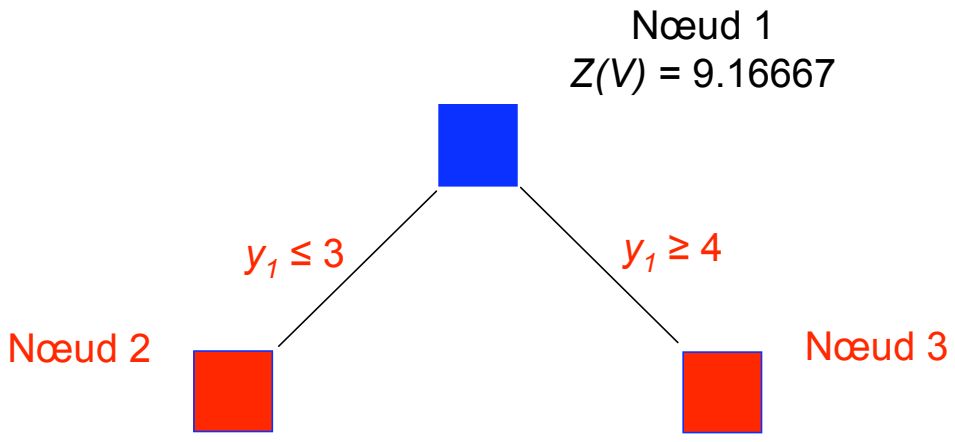
Nœud 1

Donc on commence avec un nœud (on l'appelle souvent le nœud racine).



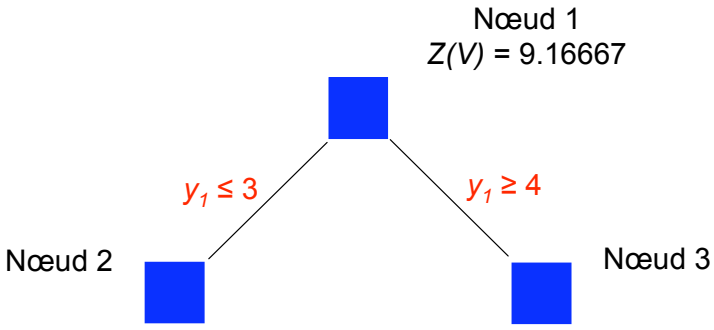
Nœud 1

Donc on commence avec un nœud (on l'appelle souvent le nœud racine).



On *divise* ce nœud en faisant *un branchement* sur une variable fractionnaire.
Ce branchement crée deux nouveaux nœuds.

Nœud 3

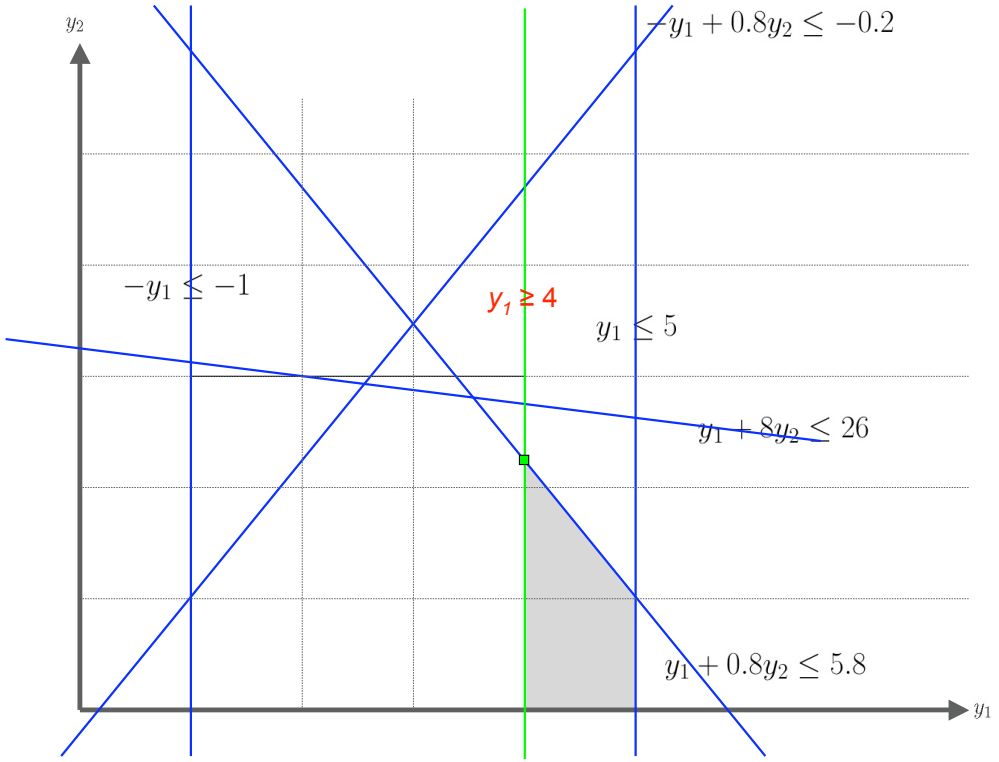


Prochainement, on résoud le programme linéaire associé au nœud 3.

La formulation associé à ce nœud est

$$
\begin{aligned}
\max \quad & y_1 + 2y_2 \\
\text{s.à} \quad & y_1 + 8y_2 \leq 26 \\
& -y_1 + 0.8y_2 \leq 0.2 \\
& y_1 + 0.8y_2 \leq 5.8 \\
& y_1 \leq 5 \\
& -y_1 \leq -1 \\
& y_1, y_2 \geq 0 \\
& y_1 \geq 4
\end{aligned}
$$

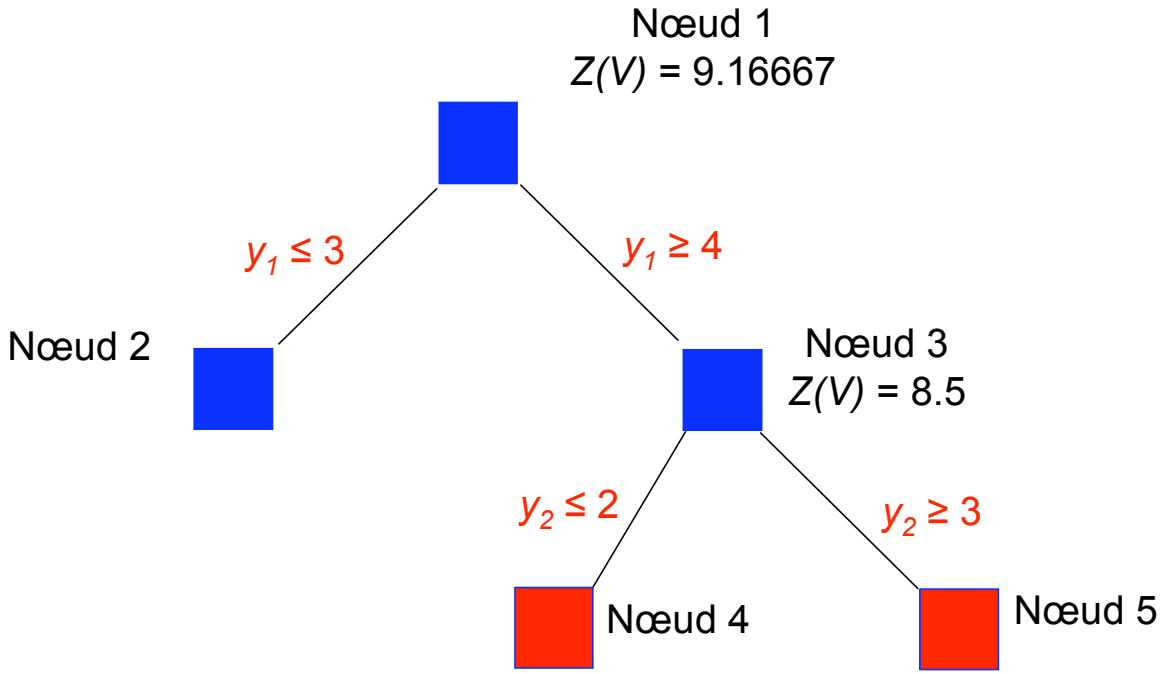
Nœud 3



La solution optimale de ce programme linéaire est $y_1 = 4$, $y_2 = \frac{9}{4} = 2\frac{1}{4}$, avec valeur objective de $8\frac{1}{2}$.

Nœud 3

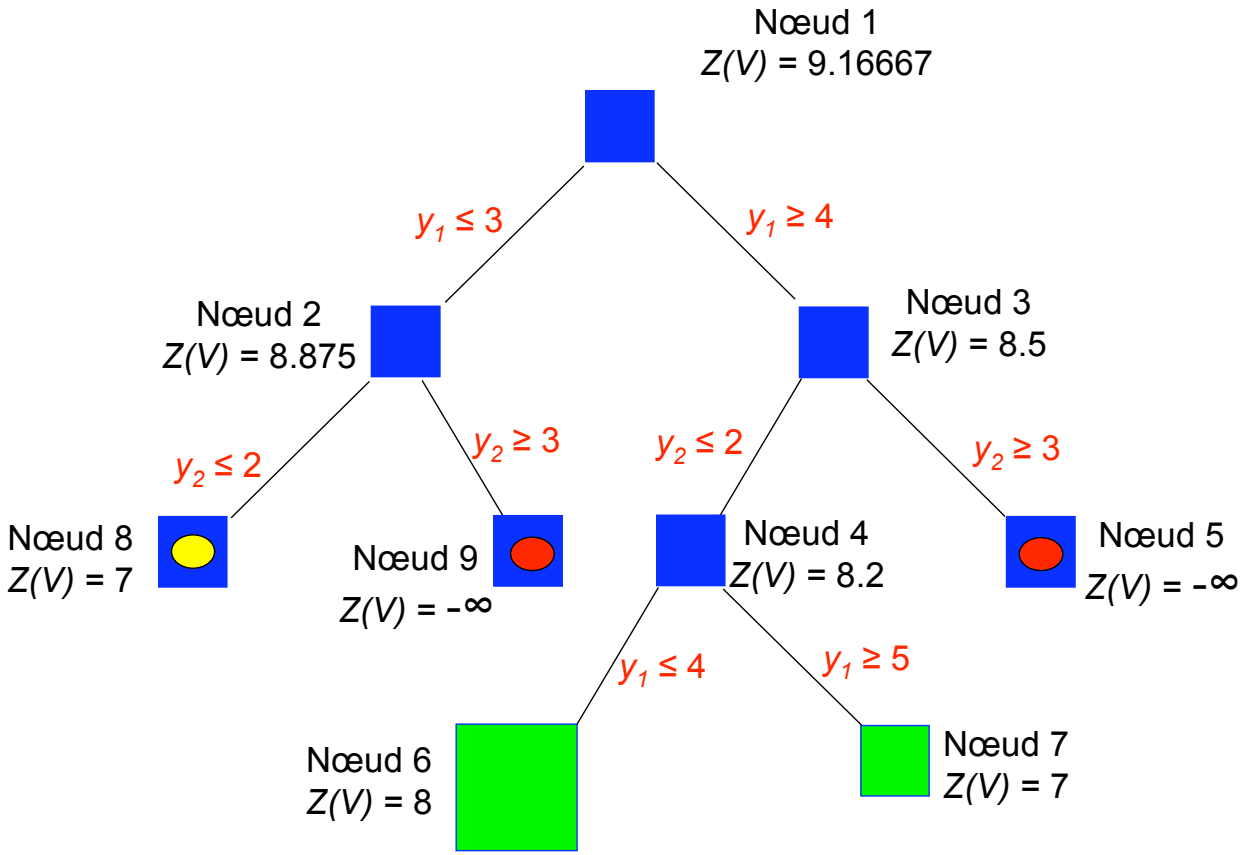
Donc on fait divise nœud 3 dans deux nouveaux nœuds : nœud 4 et nœud 5.



Continuation

On pourrait continuer jusqu'à ce que toutes les possibilités soient énumerées ou coupées.

L'algorithme générait alors la l'arbre suivant.



A souvenir

- Programmation en nombres entiers
- Modélisation et résolution des problèmes en nombres entiers
- Relaxations linéaires des problèmes en nombres entiers:
  - Ils sont des programmes linéaires dont la solution optimale définit une borne supérieure sur la valeur optimale du problème original.
  - Très important pour l'algorithme branch-and-bound
- Algorithme branch-and-bound: comment ça marche
  - division des nœuds
  - coupage des nœuds