M4202Cip – Recherche opérationnelle (V)
Procédure de Séparation et Évaluation

bruno.colombel@univ-amu.fr

IUT d'Aix-Marseille
Site d'Arles
DUT Informatique

2019–2020

Principes généraux

Exemple

Programmation linéaire en nombres entiers

Sommaire

Principes généraux

Résolution de problèmes d'optimisation combinatoire
avec un grand nombre de solutions envisageables

Principes généraux

Résolution de problèmes d'optimisation combinatoire
avec un grand nombre de solutions envisageables

On utilise une arborescence dont

- **Sommets** : ensembles de solutions réalisables
- **Racine** : ensemble de toutes les solutions réalisables

Principes généraux

L'expansion de l'arborescence est régie selon :

1. Un principe de séparation
2. Un principe d'évaluation
3. L'utilisation d'une borne
4. Une stratégie de développement

Principe de séparation

- On partage, en fonction d'un certain critère . . .

Principe de séparation

- On partage, en fonction d'un certain critère . . .
  . . . l'ensemble des solutions réalisables contenues dans un sommet de l'arborescence...

Principe de séparation

- On partage, en fonction d'un certain critère . . .
  . . . l'ensemble des solutions réalisables contenues dans un sommet de l'arborescence...
  - ensemble qui n'est pas explicitement connu

Principe de séparation

- On partage, en fonction d'un certain critère . . .
  . . . l'ensemble des solutions réalisables contenues dans un sommet de l'arborescence...
  - ensemble qui n'est pas explicitement connu
- en sous-ensembles qui deviennent alors dans l'arborescence les fils du sommet

Principe de séparation

- On partage, en fonction d'un certain critère . . .
  . . . l'ensemble des solutions réalisables contenues dans un sommet de l'arborescence...
  - ensemble qui n'est pas explicitement connu
- en sous-ensembles qui deviennent alors dans l'arborescence les fils du sommet
- il ne faut pas perdre ni ajouter de solution

Principe de séparation

- On partage, en fonction d'un certain critère . . .
  . . . l'ensemble des solutions réalisables contenues dans un sommet de l'arborescence...
  - ensemble qui n'est pas explicitement connu
- en sous-ensembles qui deviennent alors dans l'arborescence les fils du sommet
- il ne faut pas perdre ni ajouter de solution
  - L'union des sous-ensembles associés au fils du sommet doit être égale à l'ensemble associé au sommet

Principe de séparation

- Pour connaître la solution optimale

Principe de séparation

- Pour connaître la solution optimale
  - Il faut calculer la valeur de la fonction objectif pour toutes les feuilles non vides de l'arborescence
  - Ça fait beaucoup de feuilles !

Principe de séparation

- Pour connaître la solution optimale
  - Il faut calculer la valeur de la fonction objectif pour toutes les feuilles non vides de l'arborescence
  - Ça fait beaucoup de feuilles !
- On peut améliorer la méthode pour éviter l'examen de certaines branches

Principe de séparation

- Pour connaître la solution optimale
  - Il faut calculer la valeur de la fonction objectif pour toutes les feuilles non vides de l'arborescence
  - Ça fait beaucoup de feuilles !
- On peut améliorer la méthode pour éviter l'examen de certaines branches
- On ne développera pas un sommet
  - Lorsqu'on pourra montrer qu'il ne contient pas la solution optimale
  - Lorsqu'on sait résoudre directement le problème correspondant à ce sommet

Principe d'évaluation

Pour un problème d'optimisation où on maximise un objectif :

Principe d'évaluation

Pour un problème d'optimisation où on maximise un objectif :

- Borne : valeur que l'on sait atteindre pour l'objectif par une solution réalisable (minorant du maximum)

Principe d'évaluation

Pour un problème d'optimisation où on maximise un objectif :

- Borne : valeur que l'on sait atteindre pour l'objectif par une solution réalisable (minorant du maximum)
- Évaluation d'un sommet
  - Détermination d'un majorant de l'ensemble des valeurs de l'objectif des solutions de ce sommet
  - On saura alors qu'on ne pourra faire mieux que la valeur de l'évaluation
  - Elle est dite exacte lorsque la valeur donnée par l'évaluation est atteinte par un élément de l'ensemble associé au sommet

Principe d'évaluation

Pour une maximisation, la borne et l'évaluation permettent de ne pas développer un sommet S

Principe d'évaluation

Pour une maximisation, la borne et l'évaluation permettent de ne pas développer un sommet S

- Si l'évaluation de S est inférieure ou égale à la borne (on ne pourra pas trouver une solution meilleure que la borne dans cette branche)

Principe d'évaluation

Pour une maximisation, la borne et l'évaluation permettent de ne pas développer un sommet S

- Si l'évaluation de S est inférieure ou égale à la borne (on ne pourra pas trouver une solution meilleure que la borne dans cette branche)
- Si l'évaluation de S est exacte et strictement supérieure à la borne
  - On a alors trouvé une meilleure solution que la borne
  - On modifie donc la borne et on se retrouve au cas précédent (évaluation de S inférieure ou égale à la borne)

Principe d'évaluation

Pour un problème d'optimisation où on minimise un objectif :

Principe d'évaluation

Pour un problème d'optimisation où on minimise un objectif :

- Borne : valeur que l'on sait atteindre pour l'objectif par une solution réalisable (majorant du minimum)

Principe d'évaluation

Pour un problème d'optimisation où on minimise un objectif :

- Borne : valeur que l'on sait atteindre pour l'objectif par une solution réalisable (majorant du minimum)
- Évaluation d'un sommet
  - Détermination d'un minorant de l'ensemble des valeurs de l'objectif des solutions de ce sommet
  - On saura alors qu'on ne pourra faire mieux que la valeur de l'évaluation
  - Elle est dite exacte lorsque la valeur donnée par l'évaluation est atteinte par un élément de l'ensemble associé au sommet

Principe d'évaluation

Pour une minimisation, la borne et l'évaluation permettent de ne pas développer un sommet S

Principe d'évaluation

Pour une minimisation, la borne et l'évaluation permettent de ne pas développer un sommet S

- Si l'évaluation de S est supérieure ou égale à la borne (on ne pourra pas trouver une solution meilleure que la borne dans cette branche)

Principe d'évaluation

Pour une minimisation, la borne et l'évaluation permettent de ne pas développer un sommet S

- Si l'évaluation de S est supérieure ou égale à la borne (on ne pourra pas trouver une solution meilleure que la borne dans cette branche)
- Si l'évaluation de S est exacte et strictement supérieure à la borne
  - On a alors trouvé une meilleure solution que la borne
  - On modifie donc la borne et on se retrouve au cas précédent (évaluation de S supérieure ou égale à la borne)

Stratégie de développement

Détermination de la façon dont on va construire l'arborescence

- Stratégie en profondeur
- Stratégie de l'évaluation la plus grande (ou plus petite)
- Stratégie en largeur

Stratégie de développement en profondeur

On descend dans les branches jusqu'à trouver un sommet qu'on peut éliminer

Stratégie de développement en profondeur

On descend dans les branches jusqu'à trouver un sommet qu'on peut éliminer

- On remonte pour redescendre dans une autre direction
- Si on connaissait l'arborescence, ce serait équivalent à un parcours en profondeur

Stratégie de développement en profondeur

On descend dans les branches jusqu'à trouver un sommet qu'on peut éliminer

- On remonte pour redescendre dans une autre direction
- Si on connaissait l'arborescence, ce serait équivalent à un parcours en profondeur

Avantages

- Minimise la place mémoire : on ne conserve que la branche explorée et non toute l'arborescence

Stratégie de développement en profondeur

On descend dans les branches jusqu'à trouver un sommet qu'on peut éliminer

- On remonte pour redescendre dans une autre direction
- Si on connaissait l'arborescence, ce serait équivalent à un parcours en profondeur

Avantages

- Minimise la place mémoire : on ne conserve que la branche explorée et non toute l'arborescence
- Minimise les accès disque si le problème est trop grand pour tenir en mémoire principale

Stratégie de l'évaluation la plus grande

- **Pour une maximisation**
  - On pourrait suspecter que les sommets qui ont l'évaluation la plus grande contiennent la solution optimale
  - Pas nécessairement justifié

Stratégie de l'évaluation la plus grande

- **Pour une maximisation**
  - On pourrait suspecter que les sommets qui ont l'évaluation la plus grande contiennent la solution optimale
  - Pas nécessairement justifié
- Stratégie de type « meilleur d'abord »

Stratégie de l'évaluation la plus grande

- **Pour une maximisation**
  - On pourrait suspecter que les sommets qui ont l'évaluation la plus grande contiennent la solution optimale
  - Pas nécessairement justifié
- Stratégie de type « meilleur d'abord »
- Gestion de l'arborescence plus difficile et qui consomme plus d'espace mémoire

Stratégie en largeur

- On évalue tous les sommets à un niveau donné avant de descendre
- Peu utilisé
  - Rarement efficace
  - Très grande occupation mémoire

Sommaire

Exemple

$$\min z = -x_1 - x_2 - x_3 - x_4 + 3x_1x_2 - 3x_1x_3 + x_1x_4 + 2x_2x_3 - 2x_2x_4 + 2x_3x_4$$

avec $x_1, x_2, x_3, x_4 \in \{0; 1\}$.

Exemple

$$\min z = -x_1 - x_2 - x_3 - x_4 + 3x_1x_2 - 3x_1x_3 + x_1x_4 + 2x_2x_3 - 2x_2x_4 + 2x_3x_4$$

avec $x_1, x_2, x_3, x_4 \in \{0; 1\}$.

On fait les choix suivant pour la procédure arborescente :

- on sépare sur une variable x_i que l'on fixe à 1 ou à 0.
- on prend les variables dans l'ordre x_1, x_2, x_3 , x_4

Exemple

- Pour le minorant m de z on prend :
  - la valeur constituée par l'ensemble des variables fixées

Exemple

- Pour le minorant m de z on prend :
  - la valeur constituée par l'ensemble des variables fixées
  - $+$ la somme des coefficients négatifs relatifs aux variables libres restantes

Exemple

- Pour le minorant m de z on prend :
  - la valeur constituée par l'ensemble des variables fixées
  - $+$ la somme des coefficients négatifs relatifs aux variables libres restantes

Si $x_1 = 1$ et $x_2 = 0$, alors

$$\begin{aligned} z &= -1 - x_3 - x_4 - 3x_3 + x_4 + 2x_3x_4 \\ &= -1 - 4x_3 + 2x_3x_4 \geqslant -5 \qquad \text{et } m = -5 \end{aligned}$$

Exemple

- Pour le minorant m de z on prend :
  - la valeur constituée par l'ensemble des variables fixées
  - $+$ la somme des coefficients négatifs relatifs aux variables libres restantes

Si $x_1 = 1$ et $x_2 = 0$, alors

$$\begin{aligned} z &= -1 - x_3 - x_4 - 3x_3 + x_4 + 2x_3x_4 \\ &= -1 - 4x_3 + 2x_3x_4 \geqslant -5 \qquad \text{et } m = -5 \end{aligned}$$

Si $x_1 = 1$ et $x_2 = 1$, alors

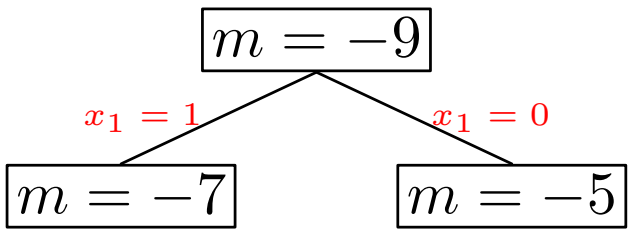
$$\begin{aligned} z &= -1 - 1 - x_3 - x_4 + 3 - 3x_3 + x_4 + 2x_3 - 2x_4 + 2x_3x_4 \\ &= 1 - 2x_3 - 2x_4 + 2x_3x_4 \geqslant -3 \qquad \text{et } m = -3 \end{aligned}$$

Exemple

La borne inférieure est calculé à chaque fois que l'on est sur une feuille de l'arbre de recherche

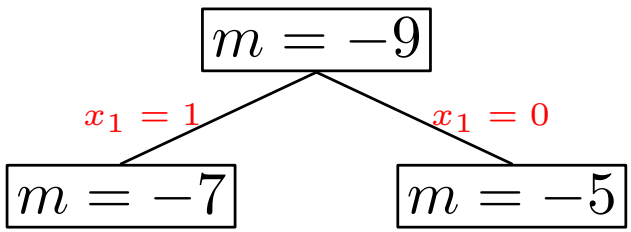
Exemple

La borne inférieure est calculé à chaque fois que l'on est sur une feuille de l'arbre de recherche



Exemple

La borne inférieure est calculé à chaque fois que l'on est sur une feuille de l'arbre de recherche

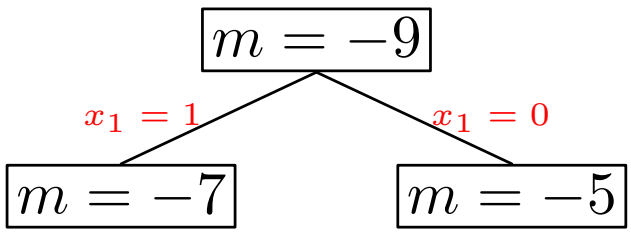


Le choix de la variable à « examiner » est arbitraire ;

- influence importante sur le nombre de nœuds à examiner

Exemple

La borne inférieure est calculé à chaque fois que l'on est sur une feuille de l'arbre de recherche



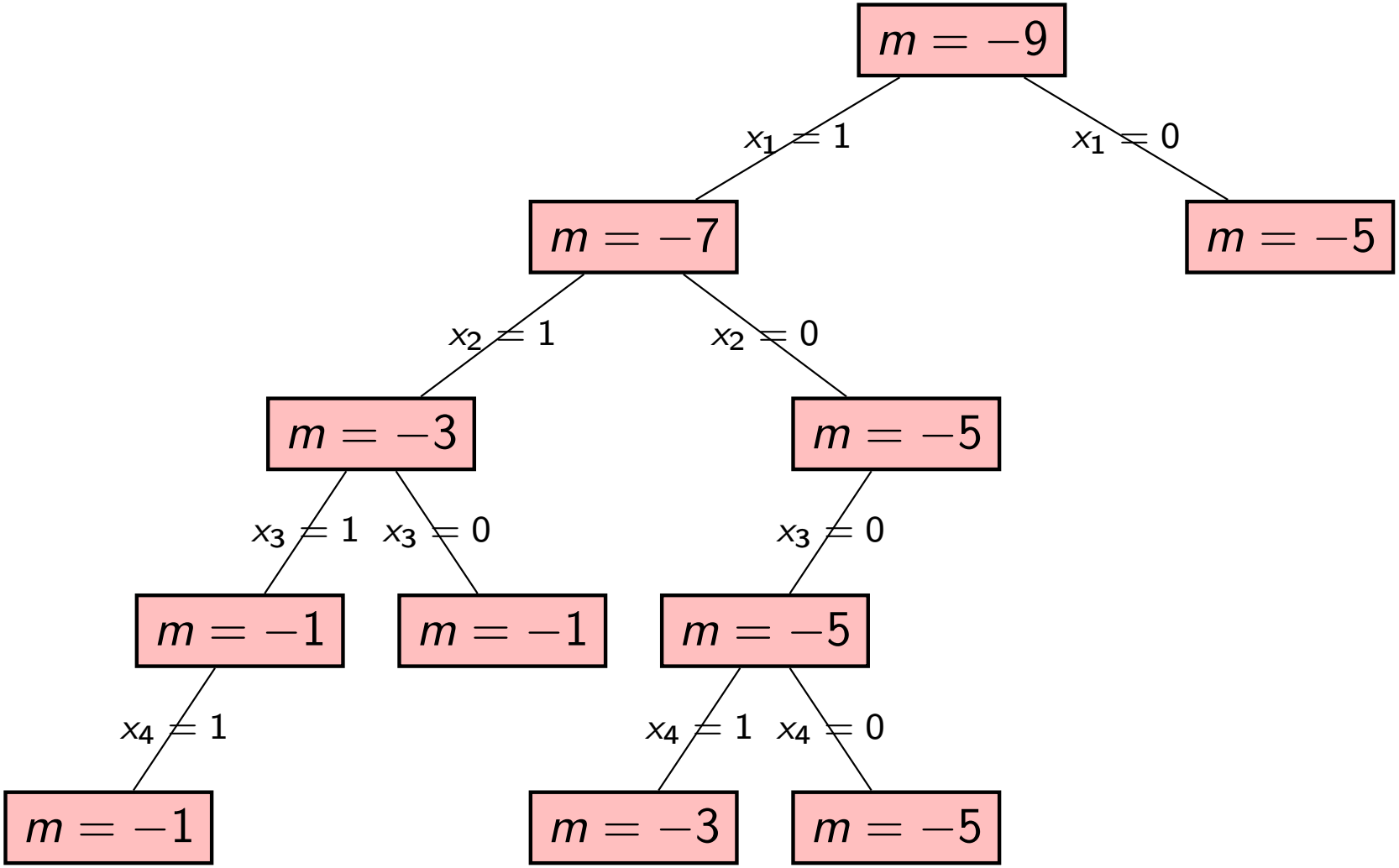
Le choix de la variable à « examiner » est arbitraire ;

- influence importante sur le nombre de nœuds à examiner

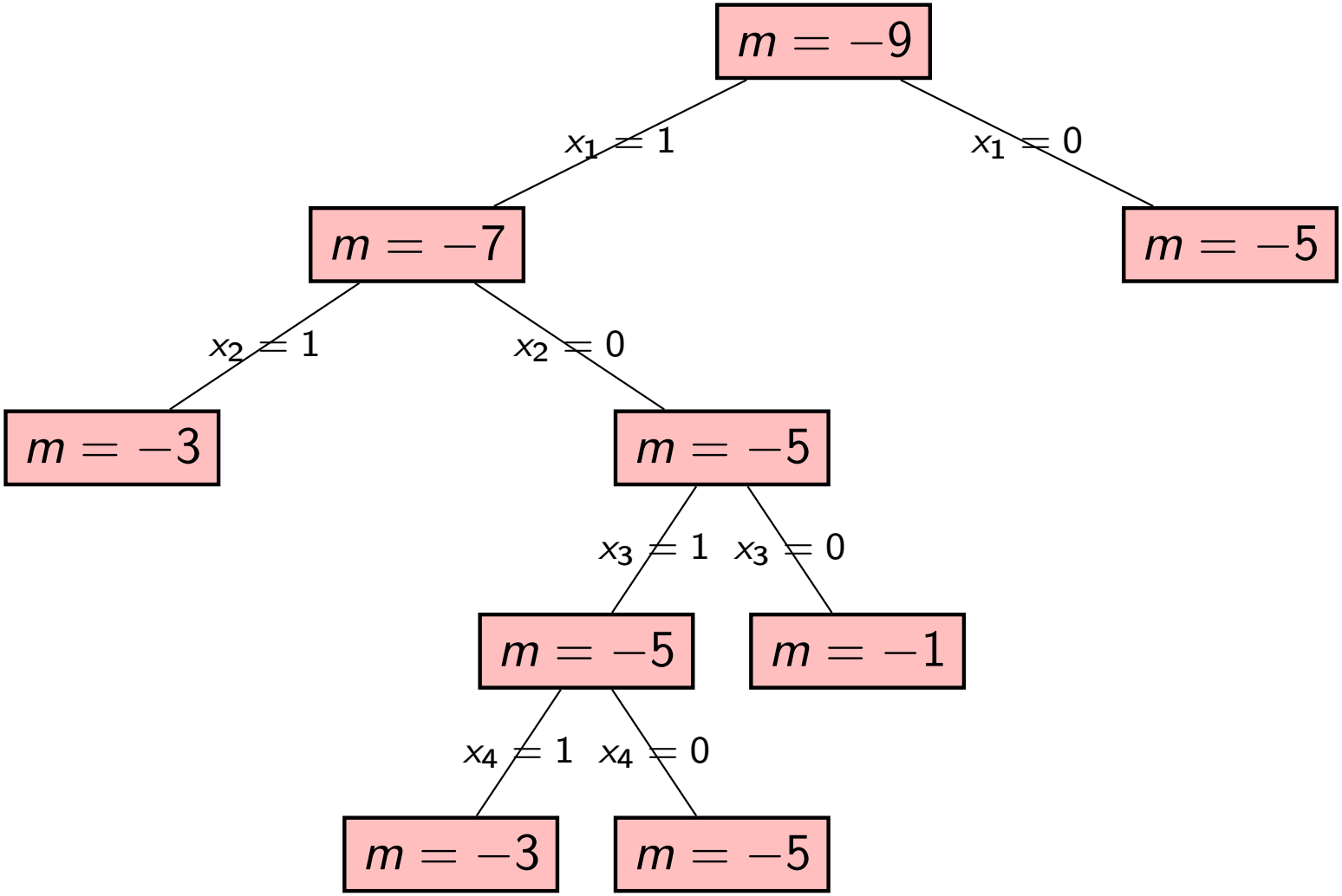
Feuille de TD

- Principes généraux

Exemple : recherche en profondeur d'abord



Exemple : recherche en meilleur d'abord



Procédures arborescentes en minimisation - résumé

- **Les bornes**
  - m : facile à calculer, calculée en chaque nœud de l'arbre
  - M : obtenue aux feuilles de l'arbre ou calculée par un algo simple

Procédures arborescentes en minimisation - résumé

- **Les bornes**
  - m : facile à calculer, calculée en chaque nœud de l'arbre
  - M : obtenue aux feuilles de l'arbre ou calculée par un algo simple
- **Séparation**
  - On énumère, on divise le pb ou sous-pb de plus en plus petits

Procédures arborescentes en minimisation - résumé

- **Les bornes**
  - m : facile à calculer, calculée en chaque nœud de l'arbre
  - M : obtenue aux feuilles de l'arbre ou calculée par un algo simple
- **Séparation**
  - On énumère, on divise le pb ou sous-pb de plus en plus petits
- **Élagage** de l'arbre de recherche grâce aux bornes m et M
  - Si $m \geqslant M$ en un nœud, on coupe ce nœud

Sommaire

PLNE

Programmation Linéaire en Nombres Entiers (PLNE)

- Variables de décision discrètes (entiers, booléens $\{0, 1\}$)
- Choix d'une bonne formulation souvent difficile
- Pas de méthode générale efficace de résolution

Branch & Bound

PLNE

Programmation Linéaire en Nombres Entiers (PLNE)

- Variables de décision discrètes (entiers, booléens $\{0, 1\}$)
- Choix d'une bonne formulation souvent difficile
- Pas de méthode générale efficace de résolution

Branch & Bound

- Énumération implicite : éliminer a priori des solutions
- Détecter que des solutions sont « mauvaises » ou irréalisables sans les évaluer explicitement

Branch & Bound pour les PLNE

1. **Évaluation** : Résoudre la relaxation linéaire
   - problème de type max z : majorant de l'optimum
   - problème de type min z : minorant de l'optimum

Branch & Bound pour les PLNE

1. **Évaluation** : Résoudre la relaxation linéaire
   - problème de type $\max z$: majorant de l'optimum
   - problème de type $\min z$: minorant de l'optimum
2. **Séparation** : brancher sur une variable non entière (à choisir)
   Si $\overline{x_i}$ est une valeur fractionnaire de la relaxation :
   - $x_i \leqslant \lfloor \overline{x_i} \rfloor$ et $x_i \geqslant \lfloor \overline{x_i} \rfloor + 1$
   - 2 sous problèmes

Branch & Bound pour les PLNE

1. **Évaluation** : Résoudre la relaxation linéaire
   - problème de type $\max z$: majorant de l'optimum
   - problème de type $\min z$: minorant de l'optimum
2. **Séparation** : brancher sur une variable non entière (à choisir)
   Si $\overline{x_i}$ est une valeur fractionnaire de la relaxation :
   - $x_i \leqslant \lfloor \overline{x_i} \rfloor$ et $x_i \geqslant \lfloor \overline{x_i} \rfloor + 1$
   - 2 sous problèmes
3. On coupe une branche si
   - La relaxation linéaire n'a pas de solution
   - la relaxation linéaire donne une solution entière