Algorithmes d’optimisation

Pr. Faouzia Benabbou (faouzia.benabbou@univh2c.ma)
Département de mathématiques et Informatique

Master Data Science & Big Data

2024-2025

FACULTE DES SCIENCES BEN M'SICK
UNIVERSITE HASSAN II DE CASABLANCA

Plan du Module: Algorithmes d'optimisation

01 Introduction

02 Rappels mathématiques

03 Algorithmes d'Optimisation Sans Contraintes

04 Algorithmes d'Optimisation avec Contraintes

05 Programmation Linéaire

06 Méthodes Optimisation combinatoire exacte

07 Méthodes Optimisation combinatoire approchées

Les algorithmes d’optimisation

Classes des algorithmes

Optimisation

Sans contraintes

Méthodes Analytiques

Méthodes numériques

- Méthode sans gradient
- Méthode de descente de gradient:
  - ✓ Méthode de gradient à pas fixe et optimal
  - ✓ Méthode de gradient conjugué
  - ✓ Méthode de Newton
  - ✓ Méthodes du gradient quasi-newton

Sous contrainte

- Méthodes générales de résolutions de PCE, PCI
- Programmation linéaire
- Optimisation combinatoire
  - ✓ Méthodes Exactes
  - ✓ Méthodes approchées
- Algorithmes multi-objectifs

Optimisation combinatoire

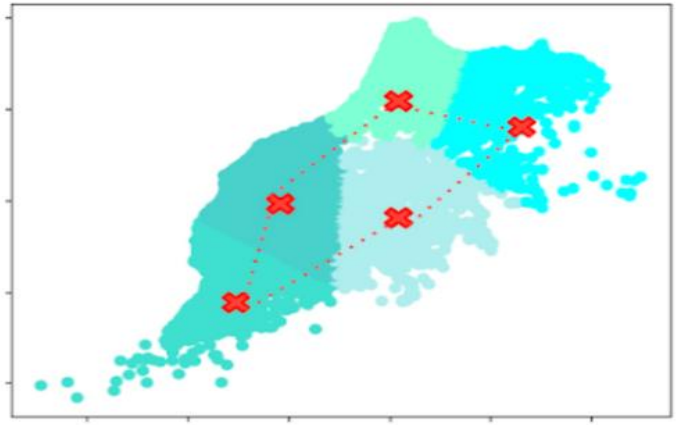
- **Définition**. L'optimisation combinatoire est une branche de l’optimisation qui consiste à trouver une solution optimale parmi un **ensemble fini** de solutions possibles, pour un problème donné.
- L’espace de recherche est fini mais souvent il es très grand (exponentiel ou factoriel).
- De nombreux problèmes d'optimisation combinatoire sont **NP-difficiles**, ce qui signifie qu'ils sont très difficiles à résoudre pour des problèmes de grande taille en un temps raisonnable.
- Le temps de calcul nécessaire pour résoudre un problème **NP-difficiles** peut, dans le pire des cas, croître de manière **exponentielle** si la taille du problème est grande.

Optimisation combinatoire

- **Exemples de problèmes combinatoires.**

1. Le problème du voyageur de commerce (TSP).

- Un voyageur de commerce doit visiter n villes données en passant par chaque ville exactement une fois.
- Il commence par une ville quelconque et termine en retournant à la ville de départ.
- Les distances entre les villes étant connues, la question que l'on se pose est de savoir quel chemin il faut choisir pour minimiser la distance totale parcourue



Optimisation combinatoire

- **Exemples.**

2. Le problème de sac à dos (Knapsack).

- Étant donné plusieurs objets possédant chacun un poids et une valeur
- Sachant que le sac a un poids maximum quels objets faut-il mettre dans le sac de manière à maximiser la valeur totale sans dépasser le poids maximal autorisé pour le sac ?.

Optimisation combinatoire

- **Exemples.**

3. Le problème d'emploi du temps (Scheduling).

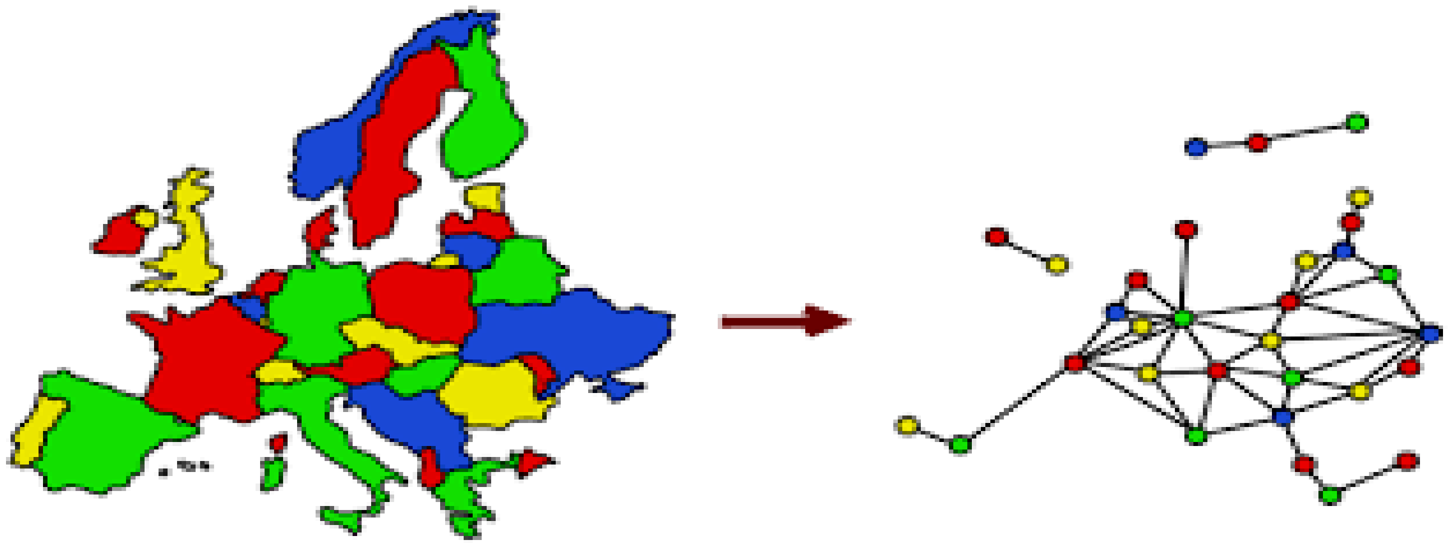
- l'objectif est de trouver la meilleure allocation possible de ressources (ex: salles, professeurs, étudiants) sur un certain nombre de périodes d temps tout en respectant des contraintes.

Optimisation combinatoire

- **Exemples.**

4. Le problème de coloriage.

- Le problème de coloration de graphes consiste à attribuer une couleur à chaque sommet d'un graphe de manière à ce que deux sommets adjacents (reliés par une arête) n'aient pas la même couleur.



Optimisation combinatoire

- **Exemples.**

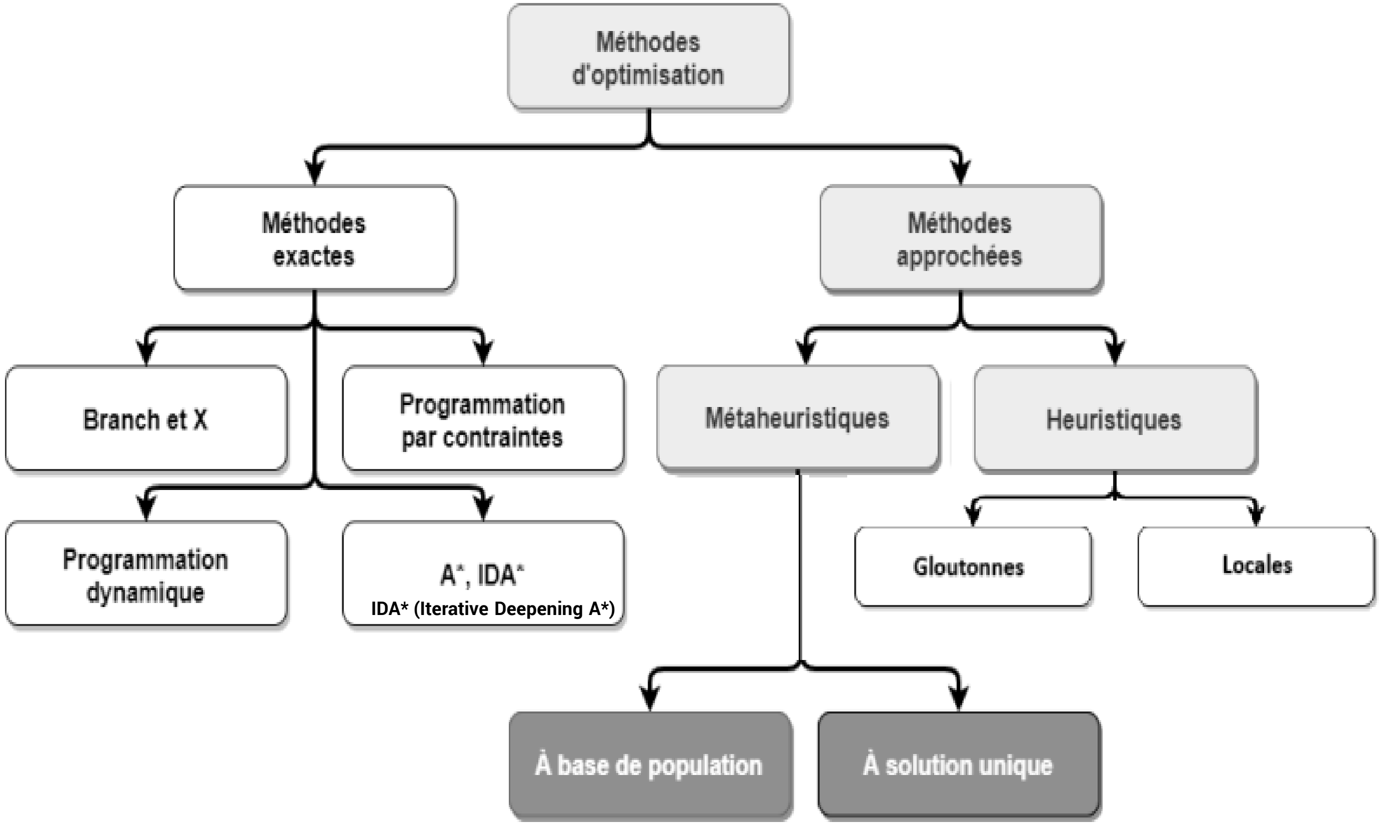
5. Le problème de sudoku.

- Le but du Sudoku est de remplir une grille de 9x9 cases avec des chiffres, afin que chaque ligne, chaque colonne et section de 3x3 cases contienne l'ensemble des chiffres de 1 à 9.

3	9				2			
		7			4	8		
		4		5		1	9	6
6	7	2	1				8	4
	3	1	9	4				
	4		7			6	1	
9	2		4		3	5	6	
		3	5		1	4		9
				6	9			

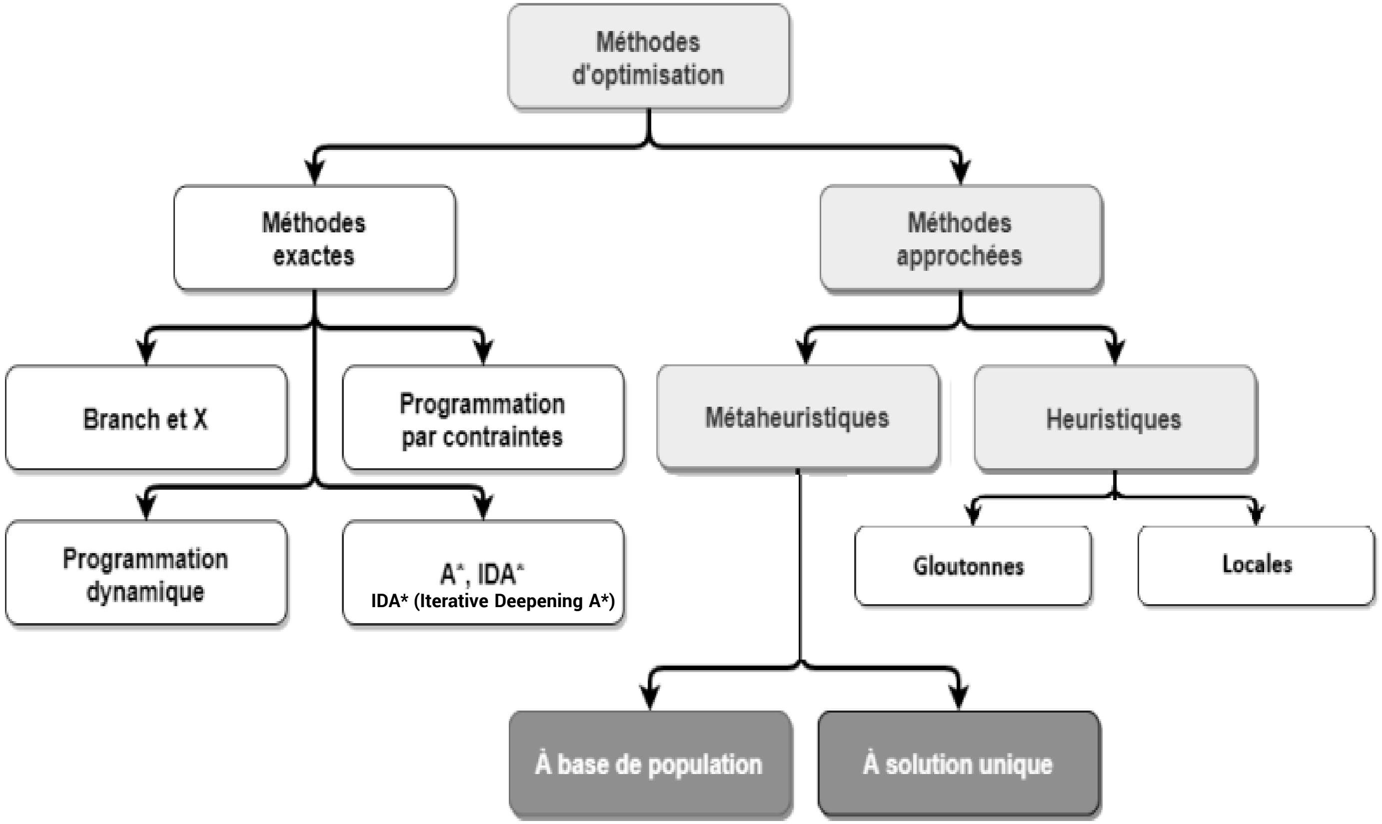
Optimisation combinatoire

- **Méthodes d’optimisation combinatoire**



Optimisation combinatoire

- **Méthodes d’optimisation combinatoire**



Optimisation combinatoire

- **Méthodes exactes**
  - **Définition.** Les méthodes combinatoires exactes sont des techniques algorithmiques conçues pour résoudre des problèmes d'optimisation en explorant de manière **exhaustive** l'ensemble, souvent très vaste mais fini, des solutions possibles, afin de garantir une solution **globale** (exacte).
  - Elles sont souvent utilisées lorsque les méthodes approchées **ne suffisent pas** pour répondre à certaines exigences de résolution.
    - ✓ **La garantit de Qualité** de la solution :les méthodes exactes trouvent **l’optimum global**.
    - ✓ **La Preuve** de l’optimalité: Fournissent une borne inférieure/supérieure qui encadre la solution

Optimisation combinatoire

- **Méthodes exactes.** Parmi les méthodes exactes utilisées en optimisation combinatoire on trouve:
  - Méthodes de Programmation **linéaire** en **nombres entiers** PLNE
    - ✓ Branch and Bound,
    - ✓ Branch and Cut,
    - ✓ Cutting Planes
    - ✓ Branch and Price
  - Méthodes de Programmation **dynamique**
  - Méthodes de Programmation par **contraintes** (Constraint Programming)

Optimisation combinatoire

▪ Méthodes exactes. Branch and Bound

- La méthode Branch and Bound (Séparation et Évaluation) se base sur la transformation d'un problème linéaire en nombres entiers (PLNE) en une série de **problèmes linéaires continus** par la technique de relaxation.
- Ces problèmes linéaires continu sont alors résolu à l'aide d'un algorithme standard pour les PL (comme le **simplexe**).

- **Définition.** Une relaxation d'un problème (P) est un nouveau problème (P_0) construit à partir de (P) et auquel on a retiré au moins une contrainte.

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound**

Exemple. Soit le problème suivant :

$$\boldsymbol{Probl\grave{e}me\ lin\acute{e}aire\ en\ nombres\ enti\grave{e}rs} \begin{cases} \boldsymbol{max\ x + y} \\ \boldsymbol{x \leq 2.5} \\ \boldsymbol{y \leq 2.5} \\ (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{N}^2 \end{cases}$$

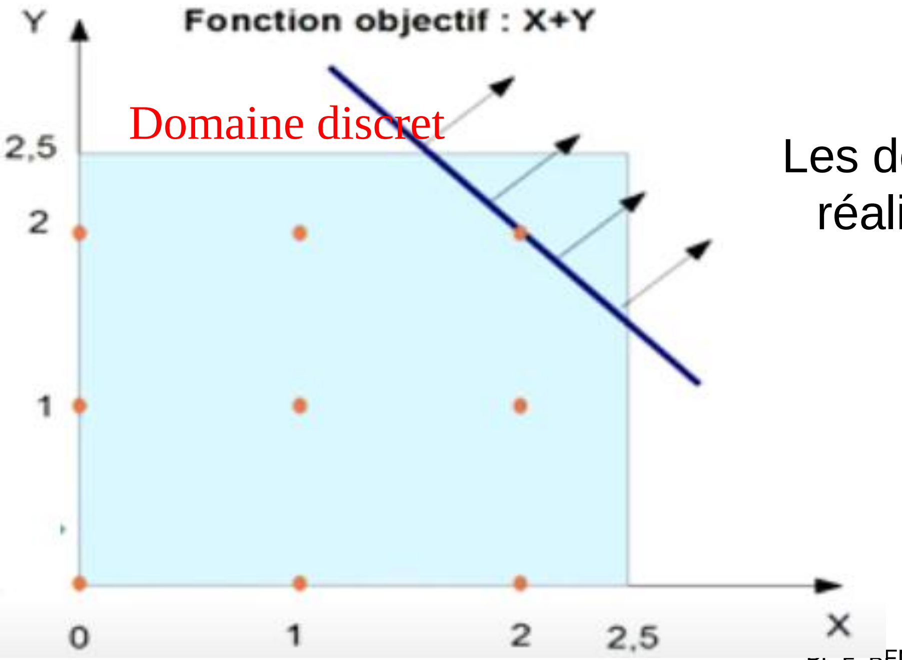
On peut passer d'un problème PLNE à un problème linéaire continue en relaxant les contraintes, le domaine de recherche devint l'ensemble des réels et **non les entiers uniquement**.

$$\boldsymbol{Probl\grave{e}me\ Lin\acute{e}aire} \begin{cases} \boldsymbol{max\ x + y} \\ \boldsymbol{x \leq 2.5} \\ \boldsymbol{y \leq 2.5} \\ (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^2 \end{cases}$$

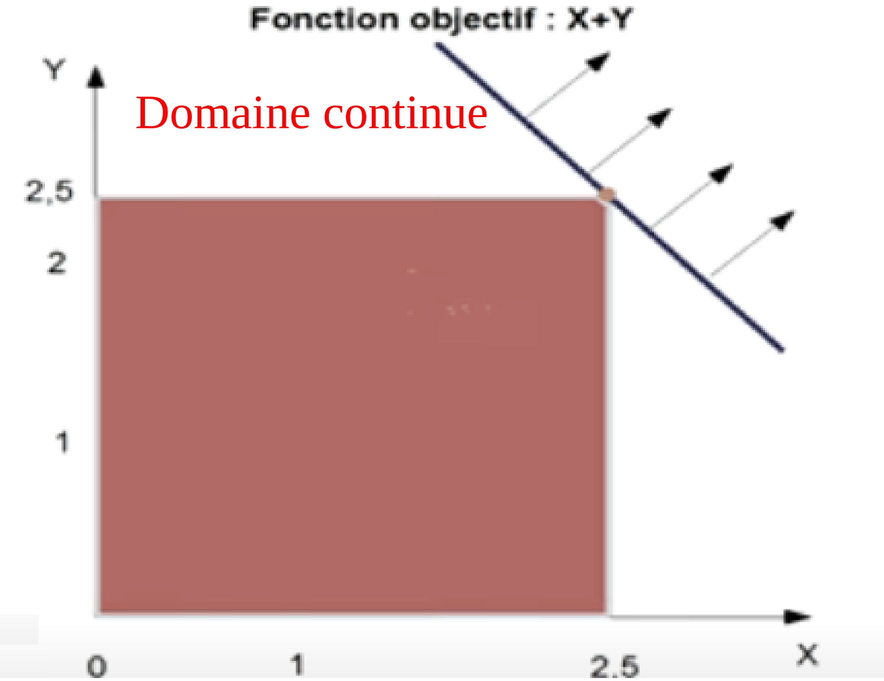
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound**

Exemple. Soit le problème $\begin{cases} \boldsymbol{max\ x+y} \\ \boldsymbol{x \le 2.5} \\ \boldsymbol{y \le 2.5} \\ (\boldsymbol{x,y}) \in \mathbb{N}^2 \end{cases}$



Les domaines réalisables



Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**
  - **Branch and Bound** repose sur deux idées principales :
    - ✓ **Branch (ramification)** : au lieu de résoudre le problème original directement, on le **divise en sous-problèmes** plus simples.
      - les sous-problèmes forment un **arbre de recherche**, dans lequel chaque nœud représente une solution.
    - ✓ **Bound (bornes)** : Une **borne** est une **estimation optimiste** de la meilleure solution que peut contenir un sous-problème (nœud) de l'arbre Branch and Bound**.** Elle est typiquement obtenue par la résolution d'une **relaxation** du sous-problème (PLNE en PL).

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**
  - Une **borne** est une **estimation** fiable de la **meilleure** solution possible pour un sous-problème donné.
    - ✓ Dans un problème de maximisation, on cherche à augmenter la valeur de la fonction objective, on utilise une borne supérieure
    - ✓ Si cette borne est plus petite que la meilleure valeur connue, on peut éliminer ce sous-problème.
    - ✓ Dans un problème de minimisation, on cherche à diminuer la valeur de la fonction objective.
    - ✓ On utilise une borne inférieure, si cette borne est plus grande que la meilleure solution admissible actuelle, on élague.

Algorithme 15. Branch and Bound(maximisation)

1. **initialisation** :

if x_0 est disponible alors **U_opt** = f(x_0), x∗ = x_0

Sinon **U_opt** ← -∞. (valeur actuelle de la meilleure solution entière trouvée).

x_opt ← ∅, k=0

Liste ← {P_0}. liste des sous-problèmes actifs à explorer.

2. Tant que Liste ≠ ∅

a) Sélectionner un sous-problème P_k dans Liste (selon une stratégie, largeur d'abord, profondeur d'abord, etc.)

b) Si P_k est **irréalisable** :

  Retirer P (Élagage par irréalisabilité)

c) Sinon Calculer le maximum global par simplexe x* de la relaxation du problème P_k : R(P_k) , U*=f(x*)

d) si **U* ≤ U_opt**:

  Retirer P (Élagage par domination)

e) Sinon si **x* est entière et U* > U_opt**:

  U_opt ← U*, x_opt ← x*

  Retirer P (Élagage par solution entière)

f) Sinon (**x* non entière et U* > U_opt**):

  Choisir une variable fractionnaire x_j*

  Créer deux sous-problèmes:

  - P_{k1} avec contrainte $x_j \leq \lfloor x_j * \rfloor$

  - P_{k2} avec contrainte $x_j \geq \lceil x_j * \rceil$

  Ajouter P_{k1} et P_{k2} à Liste

3. Retourner (x_opt, U_opt)

Algorithme 15. Branch and Bound

1. **intialisation** : Problème d'optimisation (minimisation) avec variables entières

if x0 est disponible alors L_opt = f(x0), x∗ = x0

Sinon L_opt ← +∞.

x_opt ← ∅, k=0

Liste ← $\{P_0\}$. liste des sous-problèmes actifs à explorer.

2. Tant que Liste ≠ ∅

a) Sélectionner un sous-problème P_k dans Liste (selon une stratégie, largeur d'abord, profondeur d'abord, etc.)

b) Si P_k est **irréalisable** :

  Retirer P_k (Élagage par irréalisabilité)

c) Sinon Calculer le maximum global x* de la relaxation $R(P_k)$, L*=f(x*)

d) si **L* ≥L_opt**:

  Retirer P_k (Élagage par domination)

e) Sinon si **x* est entière et L* <L_opt**:

  L_opt ← L*, x_opt ← x*

  Retirer P_k (Élagage par solution entière)

f) Sinon (**x* non entière et** L* <L_opt :

  Choisir une variable fractionnaire x_j*

  Créer deux sous-problèmes:

  - P_{k1} avec contrainte $x_j \leq \lfloor x_j^* \rfloor$

  - P_{k2} avec contrainte $x_j \geq \lceil x_j^* \rceil$

  Ajouter P_{k1} et P_{k2} à Liste

3. Retourner (x_opt, L_opt)

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

Cas	**Minimisation**
$x*$ irréalisable	Élagage par irréalisabilité
$x*$ entière **et** $L* < L\_opt$	Mise à jour de la solution optimale (meilleure borne supérieure), élagage
$x*$fractionnaire et $L* < L\_opt$	Sous-problème prometteur : **on le divise** en deux sous-problèmes (branching)
$L* \geq L\_opt$	Élagage par **domination** (le sous-problème ne peut pas améliorer la solution actuelle)

Optimisation combinatoire

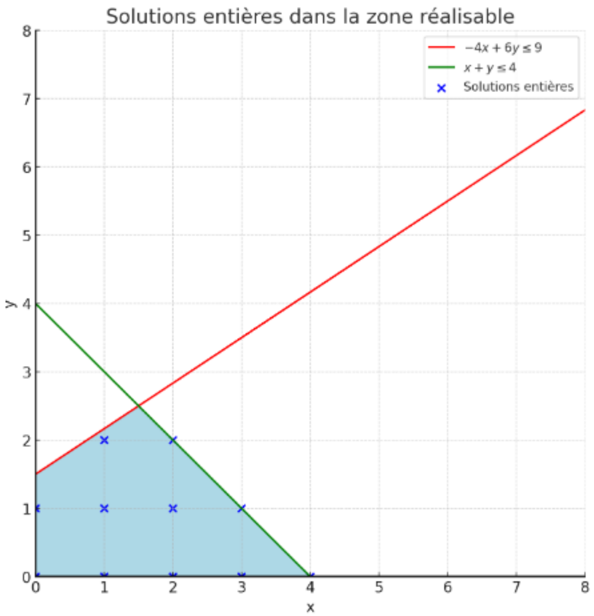
- **Méthodes exactes. Branch and Bound(B&B)**

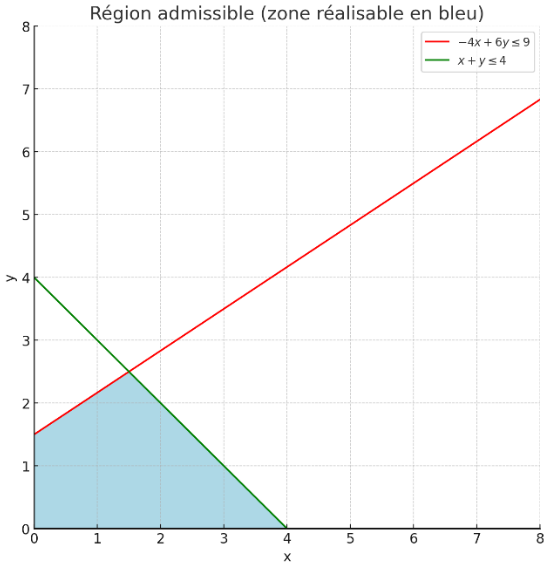
Cas	Maximisation
$x*$ irréalisable	Élagage par irréalisabilité
$x*$ entière **et** $U* > U\_opt$	Mise à jour de la solution optimale (meilleure borne supérieure), élagage
$x*$ fractionnaire et $U* > U\_opt$	Sous-problème prometteur : **on le divise** en deux sous-problèmes (branching)
$U* \leq U\_opt$	Élagage par **domination** (le sous-problème ne peut pas améliorer la solution actuelle)

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

Exemple. $$P_0 = \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \\ x_1, x_2 \in \mathbb{N} \end{cases}$$





Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 0.</u>

1.Initialisation.

L={ P_0}, x_opt, U_opt=+∝.

2.Résoudre la relaxation de P_0.

$$R(P_0)=\begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$

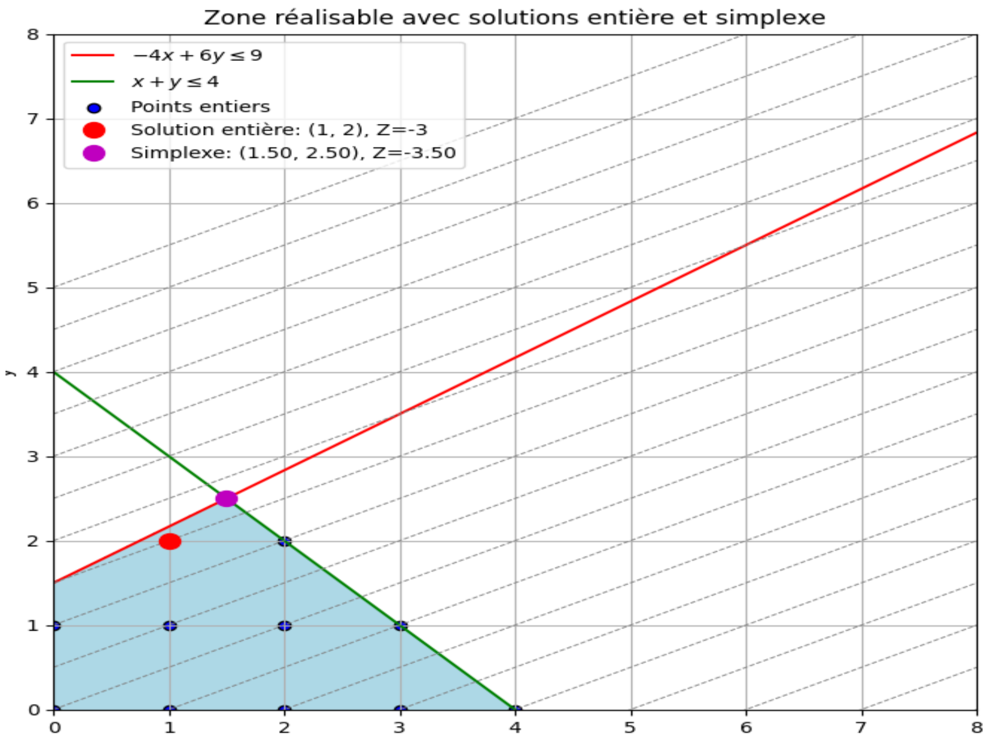
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 0.</u>

2.Résoudre la relaxation de P_0.

$$R(P_0) = \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$



Méthode graphique pour résoudre $R(P_0)$

Optimisation combinatoire

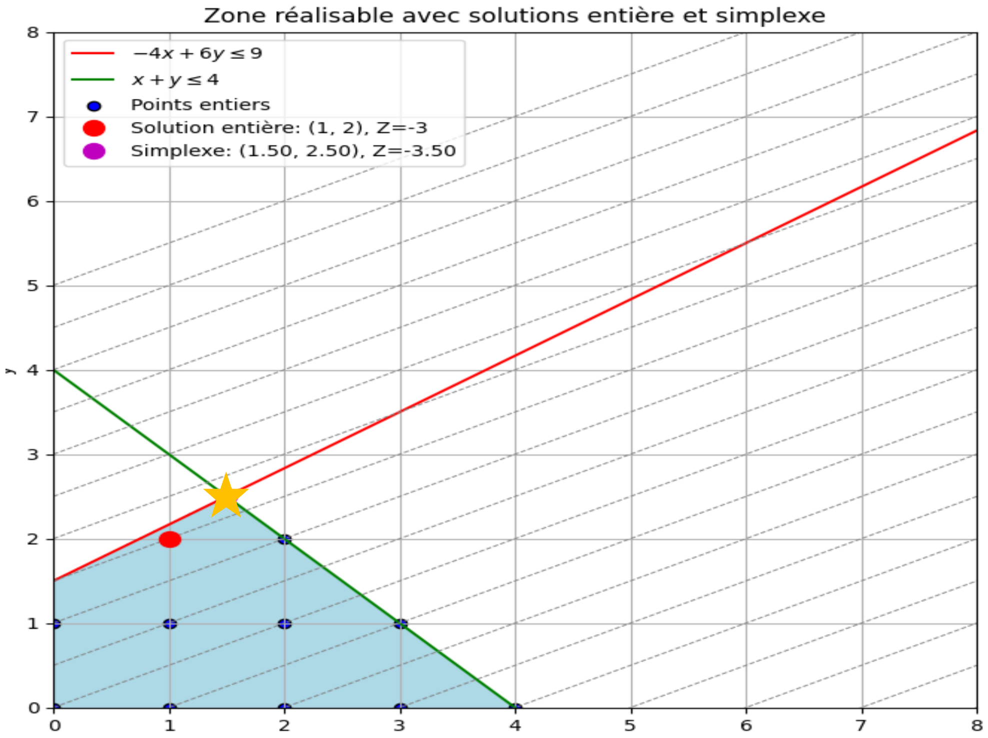
- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 0.</u>

2.Résoudre la relaxation de P0.

$$R(P_0)=\begin{cases} \boldsymbol{min\ x_1 - 2x_2} \\ \boldsymbol{-4x_1 + 6x_2 \leq 9} \\ \boldsymbol{x_1 + x_2 \leq 4} \\ \boldsymbol{x_1, x_2 \geq 0} \end{cases}$$

- ✓ La **solution optimale de R(P_0)** est : (1.50, 2.50)
- ✓ La **borne de P_0** est : U*= **-3.5**.
- ✓ **Solution n'est pas admissible pour P_0**, car ce ne sont pas des entiers
- ✓ la solution du problème relaxé nous donne une borne inférieure pour la valeur optimale du problème P_0 qui est Z_0.
- ✓ En ajoutant les contraintes d'intégrités on ne pourra pas faire mieux que cette valeur.
- ✓ On a U*< U_opt (étape f), on va brancher .



Méthode graphique pour résoudre R(P_0)

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 0.</u>

3. Choisir une variable fractionnaire : x_2= 2.50

- ✓ On va donc de brancher sur $\mathbf{x_2}$. $\mathbf{x_2 \leq 2}$ et $\mathbf{x_2 \geq 3}$
- ✓ Le choix de 2 et 3, c'est un arrondi de 2.5.de x_2 de x_op.
- ✓ On obtient deux sous problèmes P_{01} et P_{02}.

$$R(P_0)= \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$

P_{01}	P_{02}
$\min x_1 - 2x_2$	$\min x_1 - 2x_2$
s.c.	s.c.
$-4x_1 + 6x_2 \leq 9$	$-4x_1 + 6x_2 \leq 9$
$x_1 + x_2 \leq 4$	$x_1 + x_2 \leq 4$
$x_1, x_2 \geq 0$	$x_1, x_2 \geq 0$
$x_1, x_2 \in \mathbb{N}$	$x_1, x_2 \in \mathbb{N}$
$x_2 \leq 2$	$x_2 \geq 3$

L={ P_{01} , P_{02}}

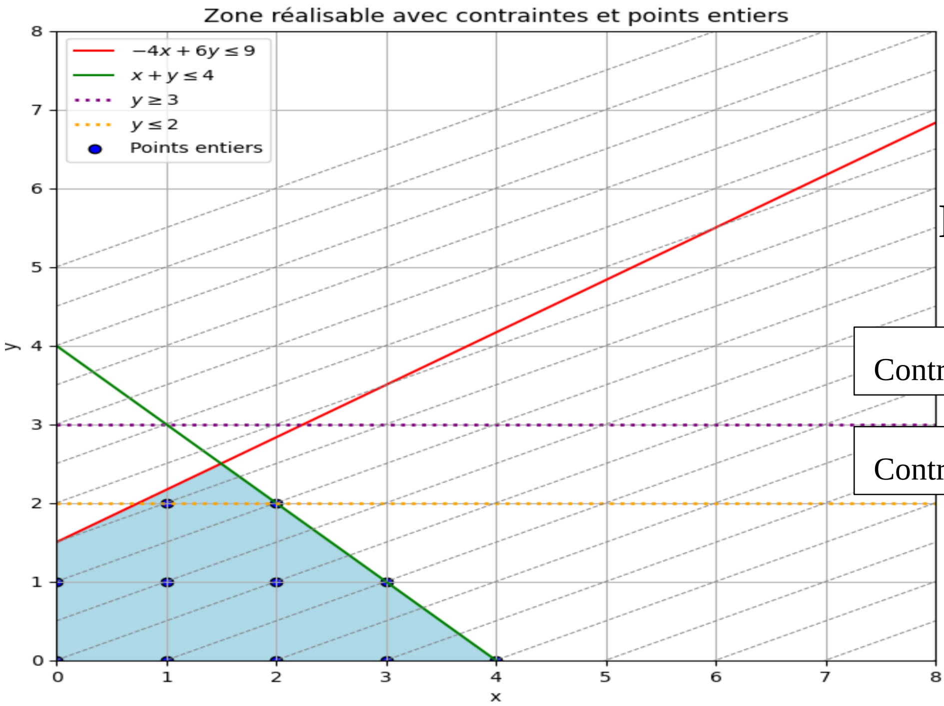
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

Itération 0.

3. Choisir une variable fractionnaire : x_2= 2.50

$$R(P_0)= \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$



Méthode graphique pour résoudre $R(P_{01})$, $R(P_{02})$

Contrainte P02, x2≥ 3

Contrainte P01, x2≤ 2

Optimisation combinatoire

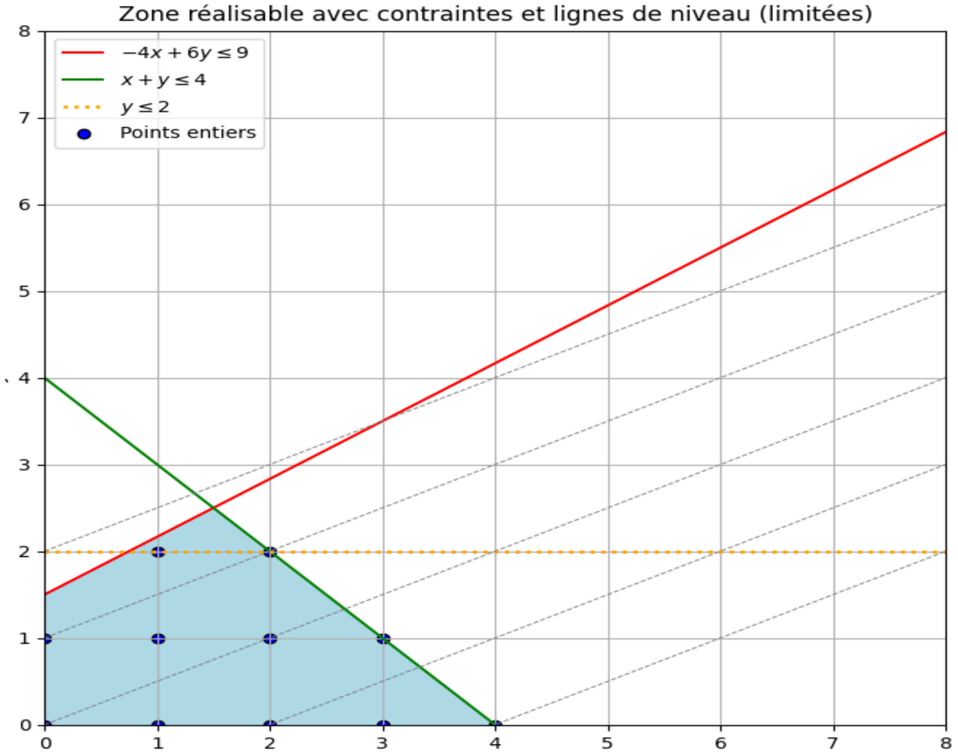
- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 1.</u>

1. **Sélectionner un sous-problème : P_{01}**

$$
\begin{array}{rcl}
\multicolumn{3}{c}{P_{01}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
\color{red}{x_2} & \color{red}{\leq} & \color{red}{2}
\end{array}
$$

On trace cette nouvelle contrainte :



Optimisation combinatoire

$$R(P_0) = \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$

- **Méthodes exactes. Branch and Bound(B&B)**

Itération 1.

2. Résoudre le problème relaxé $R(P_{01})$.

(P_{01})

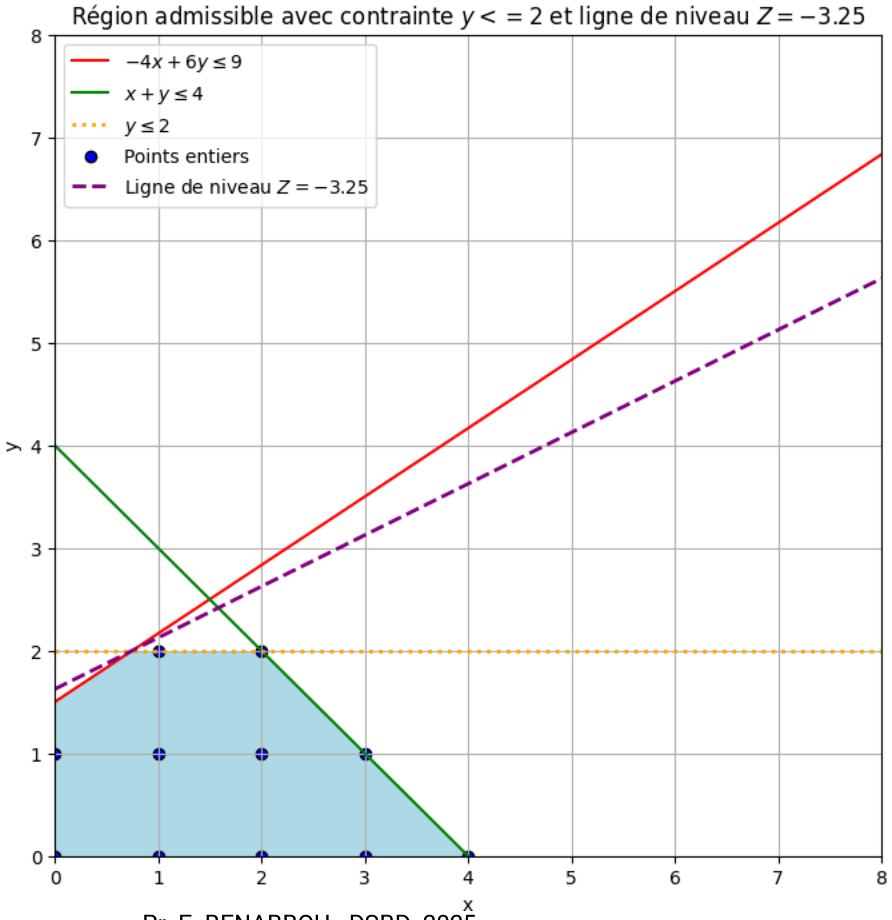
min $x_1 - 2x_2$ sous contraintes

$$\begin{aligned} -4x_1 + 6x_2 &\leq 9 \\ x_1 + x_2 &\leq 4 \\ x_1, x_2 &\geq 0 \\ x_2 &\leq 2 \\ x_1, x_2 &\in \mathbb{N}. \end{aligned}$$

$R(P_{01})$

min $x_1 - 2x_2$ sous contraintes

$$\begin{aligned} -4x_1 + 6x_2 &\leq 9 \\ x_1 + x_2 &\leq 4 \\ x_1, x_2 &\geq 0 \\ x_2 &\leq 2 \end{aligned}$$



Optimisation combinatoire

$$R(P_0) = \begin{cases} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{cases}$$

- **Méthodes exactes. Branch and Bound(B&B)**

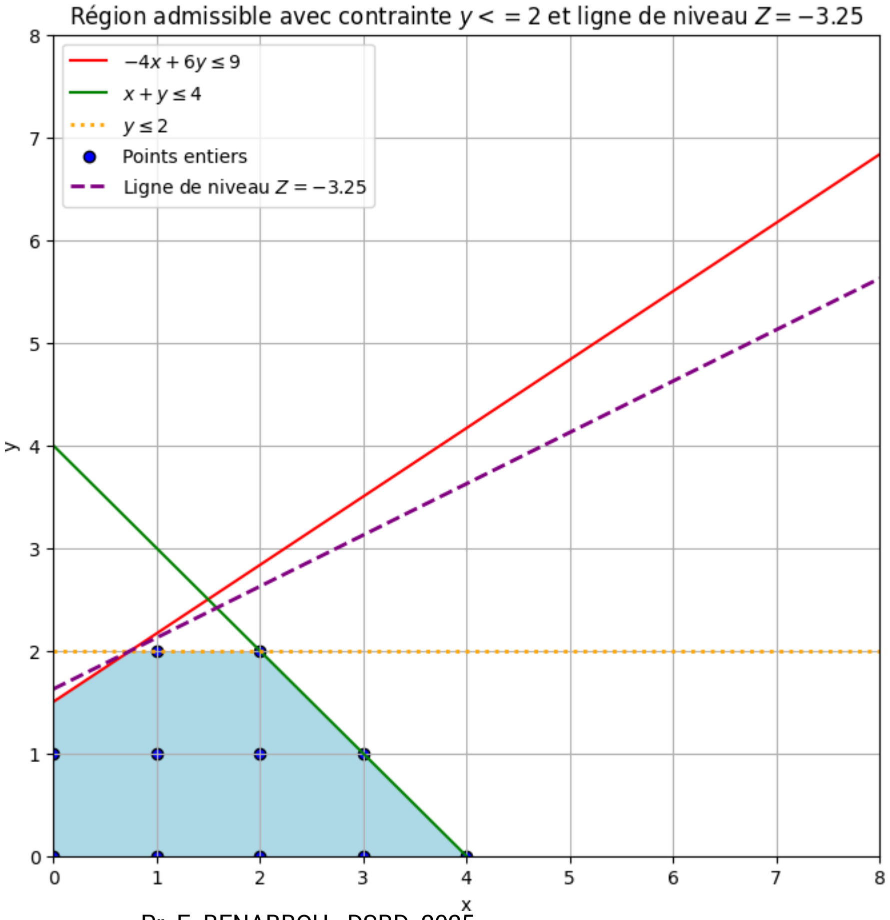
<u>Itération 1.</u>

2. Résoudre le problème relaxé R(P_{01})

Le problème $\mathbf{P_{01}}$ admet un optimum.

C'est le sommet du polyèdre qui touche la courbe de niveau de f égal à **-3,25**.

- ✓ **Solution optimale : (0.75,2)**
- ✓ **Borne pour P_{01} : L*= -3,25.**
- ✓ L*<L_opt et x_1=0,75 n'est pas entier donc on va brancher



Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 1.</u>

3. Choisir une variable fractionnaire pour brancher : x1=0.75

- ✓ On va créer deux branches sur x_1 : $\mathbf{x_1 \leq 0}$ **et** $\mathbf{x_1 \geq 1}$. On obtient deux sous problèmes : $P_{\mathbf{011}}, P_{\mathbf{012}}$.
- ✓ L={ $\mathbf{P_{011}}, \mathbf{P_{012}}$, $P_{\mathbf{02}}$}
- ✓ Comme x1 ≥0 et x1≤0 donc x_1=0.

P_{011}	P_{012}
$\min x_1 - 2x_2$	$\min x_1 - 2x_2$
s.c.	s.c.
$-4x_1 + 6x_2 \leq 9$	$-4x_1 + 6x_2 \leq 9$
$x_1 + x_2 \leq 4$	$x_1 + x_2 \leq 4$
$x_1, x_2 \geq 0$	$x_1, x_2 \geq 0$
$x_1, x_2 \in \mathbb{N}$	$x_1, x_2 \in \mathbb{N}$
$x_2 \leq 2$	$x_2 \leq 2$
$x_1 \leq 0$	$x_1 \geq 1$

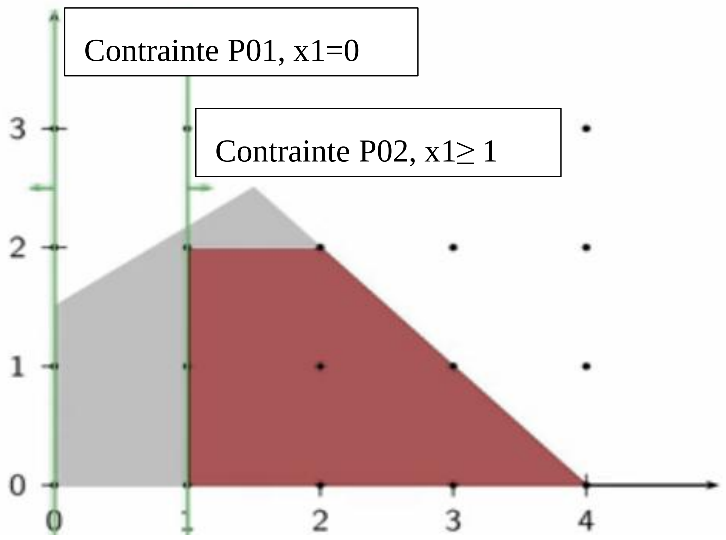
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 1.</u>

3. Choisir une variable fractionnaire pour brancher : x_1=0.75

P_{011}	P_{012}
$\min x_1 - 2x_2$	$\min x_1 - 2x_2$
s.c.	s.c.
$-4x_1 + 6x_2 \leq 9$	$-4x_1 + 6x_2 \leq 9$
$x_1 + x_2 \leq 4$	$x_1 + x_2 \leq 4$
$x_1, x_2 \geq 0$	$x_1, x_2 \geq 0$
$x_1, x_2 \in \mathbb{N}$	$x_1, x_2 \in \mathbb{N}$
$x_2 \leq 2$	$x_2 \leq 2$
$x_1 \leq 0$	$x_1 \geq 1$



L={ $\mathbf{P_{011}}$,$\mathbf{P_{012}}$, P_{02}}

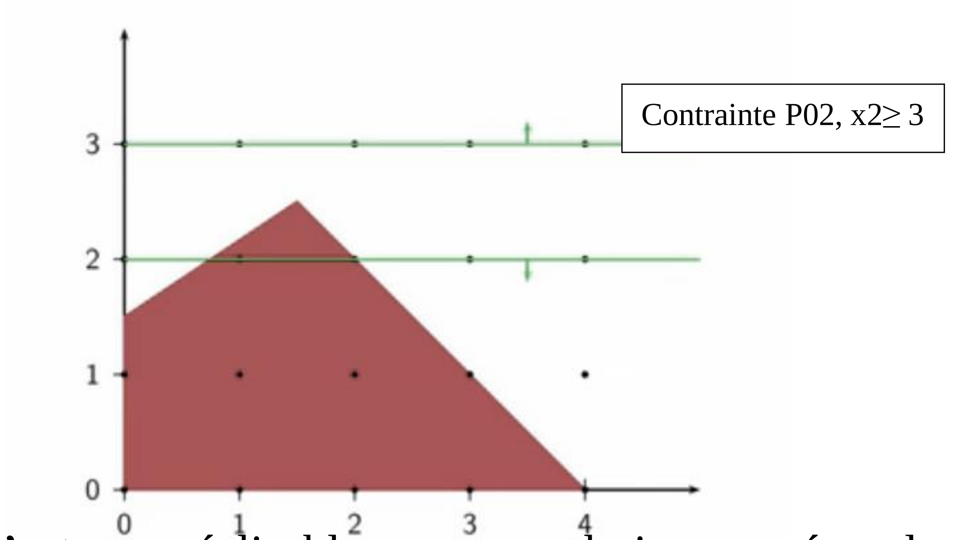
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 2.</u>

1. Sélectionner un sous-problème : P_{02}

$$
\begin{array}{lcl}
\multicolumn{3}{c}{P_{02}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
\color{red}{x_2} & \color{red}{\geq} & \color{red}{3}
\end{array}
$$



Pour le sous problème **P_{02}** où on prend $x_2 \geq 3$, n'est pas réalisable aucune solution ne réponds aux contraintes originales. On **élague** cette branche et retire P_{02} de L.

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 3.</u>

1. Sélectionner un sous-problème : P_{011}

$$
\begin{array}{rcl}
\multicolumn{3}{c}{P_{011}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
x_2 & \leq & 2 \\
x_1 & \leq & 0
\end{array}
$$

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 3.</u>

2.Résoudre le problème relaxé R(P_{011})

P_{011} **R(P_{011})**

$$
\begin{array}{rcl}
\multicolumn{3}{c}{P_{011}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
x_2 & \leq & 2 \\
x_1 & \leq & 0
\end{array}
$$

$$
\begin{array}{rcl}
\multicolumn{3}{l}{\min x_1 - 2x_2 \text{ sous contraintes}} \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_2 & \leq & 2 \\
x_1 & \leq & 0
\end{array}
$$

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 3.</u>

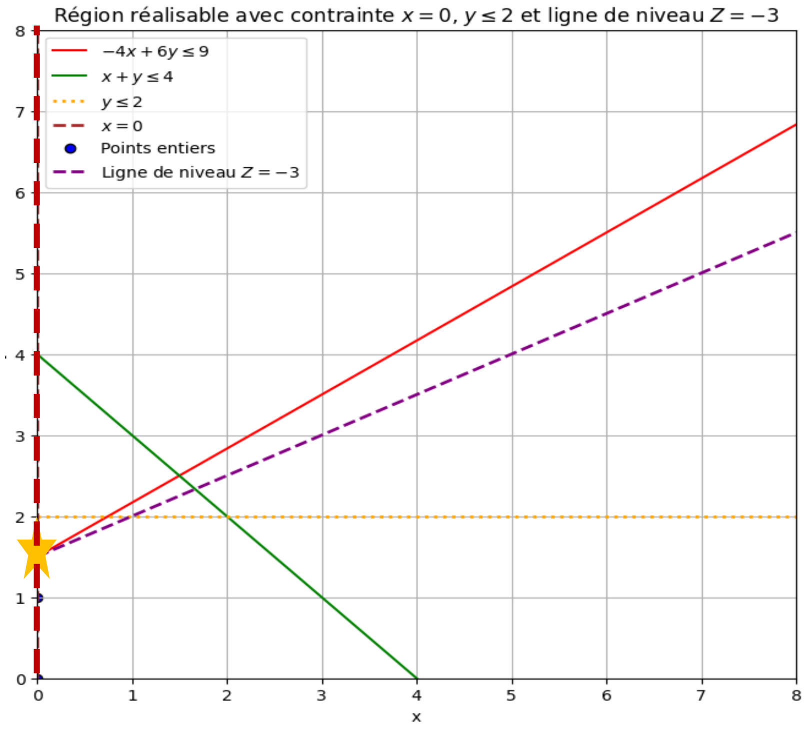
2.Résoudre le problème relaxé R(P_{011})

R(P_{011})

$$\min x_1 - 2x_2 \text{ sous contraintes}$$

$$\begin{aligned} -4x_1 + 6x_2 &\leq 9 \\ x_1 + x_2 &\leq 4 \\ x_1, x_2 &\geq 0 \\ x_2 &\leq 2 \\ x_1 &\leq 0 \end{aligned}$$

x1=0, méthode de graphe



Solution optimale : (0,1.5)

Borne pour P_{011} : L*= -3.

L*< L_opt, x_2=1,5 n'est pas entier, donc on va brancher sur x2.

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

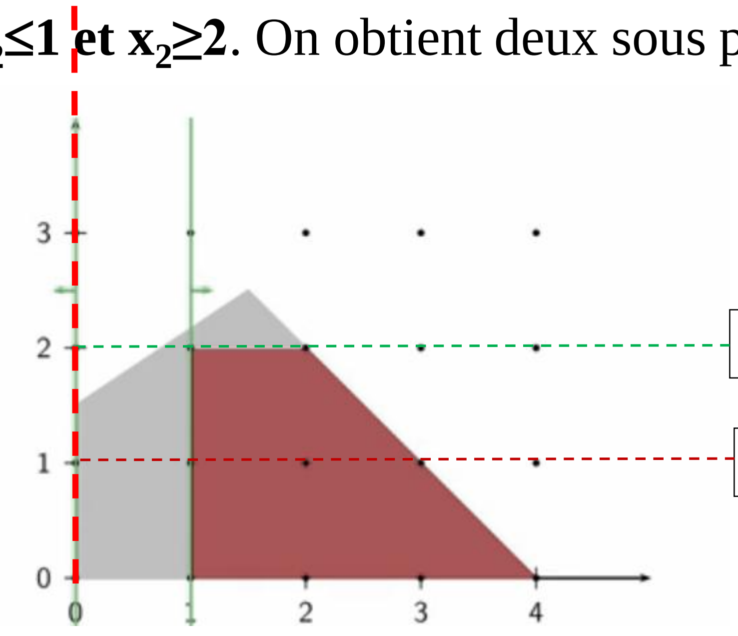
<u>Itération 3.</u>

3. Choisir une variable fractionnaire pour brancher : x_2

On va créer deux branches sur x_2 : $\mathbf{x_2 \leq 1}$ **et** $\mathbf{x_2 \geq 2}$. On obtient deux sous problèmes : $P_{\mathbf{0111}}$,$P_{\mathbf{0112}}$.

L={ $P_{\mathbf{012}}$, $P_{\mathbf{0111}}$,$P_{\mathbf{0112}}$ }

P_{0111}	P_{0112}
$\min x_1 - 2x_2$ $-4x_1 + 6x_2 \leq 9$ $x_1 + x_2 \leq 4$ $x_1, x2 \geq 0$ $x_1, x_2 \in \mathbb{N}$ $x_2 \leq 2$ $x_1 \leq 0$ $x_2 \leq 1$	$\min x_1 - 2x_2$ $-4x_1 + 6x_2 \leq 9$ $x_1 + x2 \leq 4$ $x_1, x_2 \geq 0$ $x_1, x_2 \in \mathbb{N}$ $x_2 \leq 2$ $x_1 \leq 0$ $x_2 \geq 2$



Contrainte P0112, x1=0, x2≥2

Contrainte P0111, x1=0 , x2≤1

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>**Itération 4.**</u>

1. Sélectionner un sous-problème : On choisit $\mathbf{P_{012}}$.

$$
\begin{array}{rcl}
\multicolumn{3}{c}{P_{012}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
x_2 & \leq & 2 \\
x_1 & \geq & 1
\end{array}
$$

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 4.</u>

2.Résoudre le problème relaxé R(P_{012})

P_{012} **R(P_{012})**

$$
\begin{array}{rcl}
\multicolumn{3}{c}{P_{012}} \\
\hline
\multicolumn{3}{c}{\min x_1 - 2x_2} \\
\text{s.c.} & & \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_1, x_2 & \in & \mathbb{N} \\
x_2 & \leq & 2 \\
x_1 & \geq & 1
\end{array}
$$

$$
\begin{array}{rcl}
\multicolumn{3}{l}{\min x_1 - 2x_2 \text{ sous contraintes}} \\
-4x_1 + 6x_2 & \leq & 9 \\
x_1 + x_2 & \leq & 4 \\
x_1, x_2 & \geq & 0 \\
x_2 & \leq & 2 \\
x_1 & \geq & 1
\end{array}
$$

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

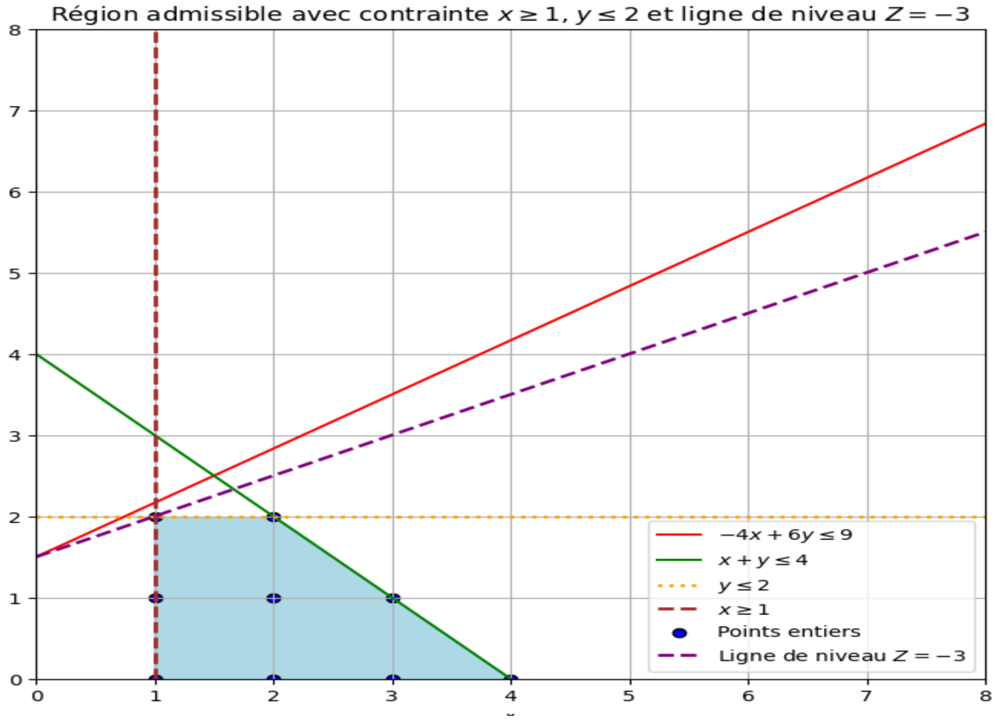
<u>Itération 4.</u>

2.Résoudre le problème relaxé $R(P_{012})$

$R(P_{012})$

$$\min x_1 - 2x_2 \text{ sous contraintes}$$

$$\begin{aligned}
-4x_1 + 6x_2 &\leq 9 \\
x_1 + x_2 &\leq 4 \\
x_1, x_2 &\geq 0 \\
x_2 &\leq 2 \\
x_1 &\geq 1
\end{aligned}$$



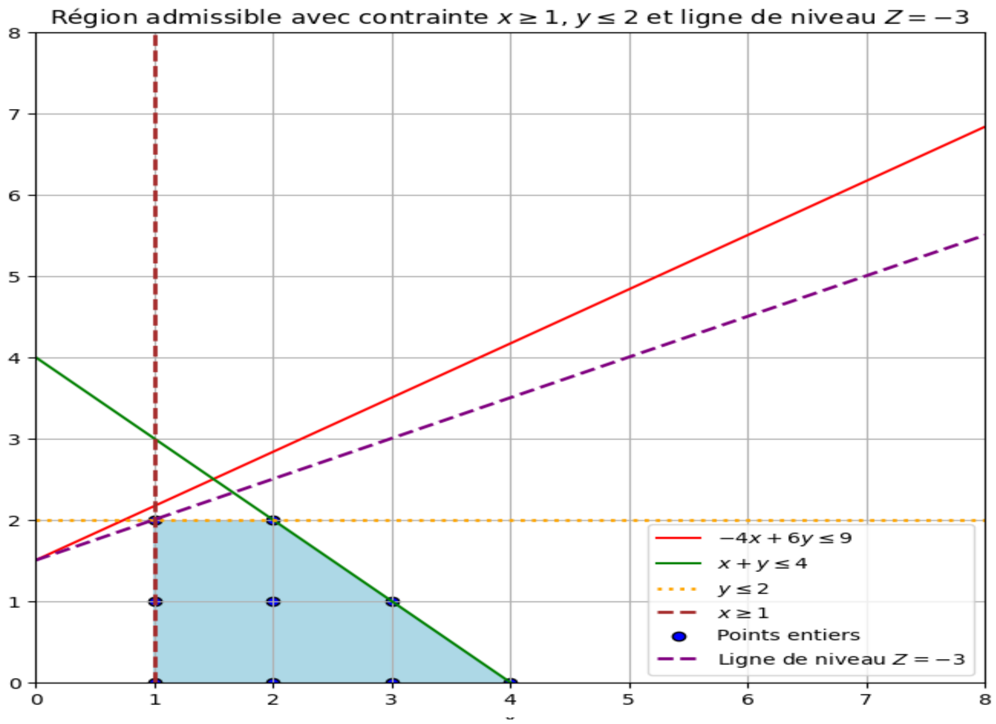
Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 4.</u>

2.Résoudre le problème relaxé R(P_{012})

- ✓ **Solution optimale : (1, 2)**
- ✓ **Borne pour P_{012} : U*= -3.**
- ✓ L*<L_opt et x* est entier donc on met à jour :
- ✓ **L_opt =L*=-3, et x_opt = (1, 2)**
- ✓ **Cette solution est admissible pour P_{012},** et nous avons résolu P_{01} et par la même occasion P_0 mais L n'est pas encore vide: L={P_{0111},P_{0112} }



Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 5.</u>

1. Choisir le problème P_{0111}

P_{0111}

$$\min x_1 - 2x_2$$
$$-4x_1 + 6x_2 \leq 9$$
$$x_1 + x_2 \leq 4$$
$$x_1, x2 \geq 0$$
$$x_1, x_2 \in \mathbb{N}$$
$$x_2 \leq 2$$
$$x_1 \leq 0$$
$$x_2 \leq 1$$

$R(P_{0111})$

$$\min x_1 - 2x_2$$
$$-4x_1 + 6x_2 \leq 9$$
$$x_1 + x_2 \leq 4$$
$$x_1, x2 \geq 0$$
$$x_2 \leq 2$$
$$x_1 \leq 0$$
$$x_2 \leq 1$$

Optimisation combinatoire

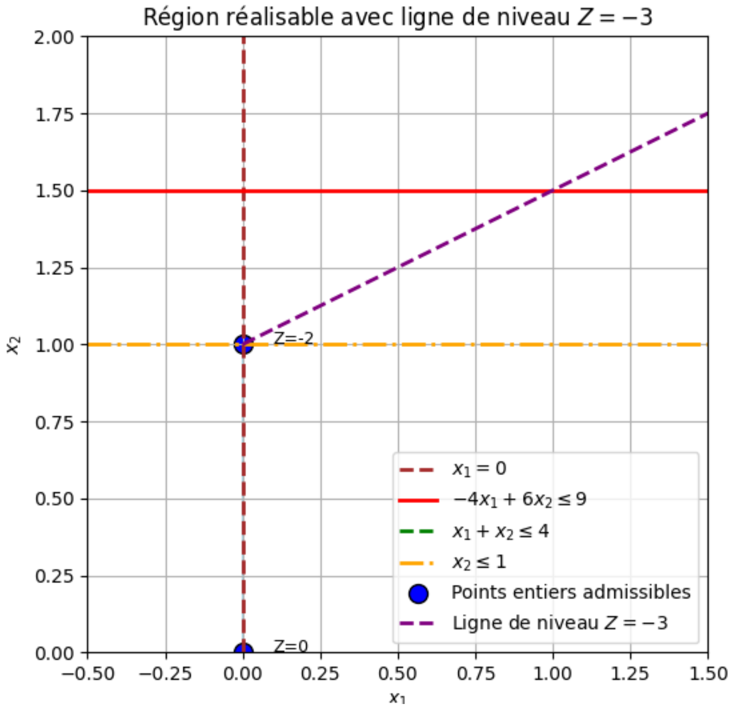
- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 5.</u>

2 . Résoudre le problème P_{0111} Dernière valeur de U_opt est -3.

$R(P_{0111})$

$$\min x_1 - 2x_2$$
$$-4x_1 + 6x_2 \leq 9$$
$$x_1 + x_2 \leq 4$$
$$x_1, x2 \geq 0$$
$$x_2 \leq 2$$
$$x_1 \leq 0$$
$$x_2 \leq 1$$



Solution optimale entière : x_opt =(0,1),

Borne de P_{0111} : L*=−2.

On a **L* > L_opt**, Retirer P_{0111} de la liste L car ça n'améliore pas la fonction objective (**Élagage par domination**).

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

Itération 6.

1.Choisir le problème P_{0112}

$$P_{0112}$$

$$\begin{gathered} \min x_1 - 2x_2 \\ -4x_1 + 6x_2 \leq 9 \\ x_1 + x_2 \leq 4 \\ x_1, x2 \geq 0 \\ x_1, x_2 \in \mathbb{N} \\ x_2 \leq 2 \\ x_1 \leq 0 \\ x_2 \geq 2 \end{gathered}$$

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**

<u>Itération 6.</u>

2.Résoudre le problème R(P_{0112})

P_{0112}	R(P_{0112})
$\min x_1 - 2x_2$	$\min x_1 - 2x_2$
$-4x_1 + 6x_2 \le 9$	$-4x_1 + 6x_2 \le 9$
$x_1 + x_2 \le 4$	$x_1 + x_2 \le 4$
$x_1, x2 \ge 0$	$x_1, x2 \ge 0$
$x_1, x_2 \in \mathbb{N}$	$x_2 \le 2$
$x_2 \le 2$	$x_1 \le 0$
$x_1 \le 0$	$x_2 \ge 2$
$x_2 \ge 2$	

- ✓ On a $\mathbf{x_1=0}$, $\mathbf{x_2=2}$, donc 12<=9 ce qui n'est pas réalisable.
- ✓ donc P_{0112} n'est pas réalisable on le retire de L.
- ✓ **L est vide** : la solution optimale est **L_opt = - 3, x_opt= (1, 2)**

Optimisation combinatoire

- **Méthodes exactes. Branch and Bound(B&B)**
  - **Avantages.**
    - ✓ Garanti une solution optimale car il trouve la solution exacte.
    - ✓ Utilise des bornes (supérieures/inférieures) pour éliminer les sous-problèmes non prometteurs réduisant ainsi l'espace de recherche en évitant l'exploration inutile.
    - ✓ Applicable à divers problèmes (PLNE, voyageur de commerce, etc.).
    - ✓ Performant pour les petits/moyens problèmes
  - **Limites.**
    - ✓ Complexité exponentielle **:** Le nombre de nœuds peut exploser avec la taille du problème (*NP-difficile*) il peut devenir Inadapté aux très grands problèmes (milliers de variables).
    - ✓ Gourmand en mémoire : Nécessite de stocker l'arbre de recherche, ce qui peut être coûteux.