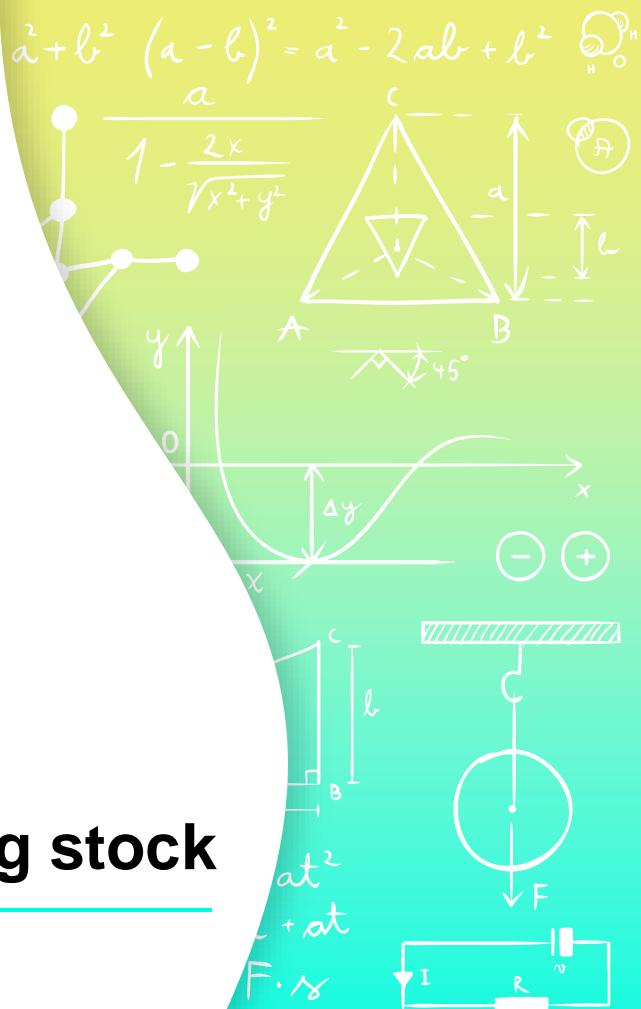


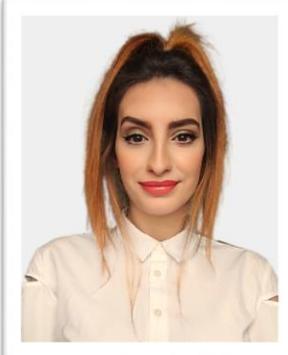
# PRÉSENTATION

Étude d'un algorithme  
de génération de colonnes  
pour les problèmes de  
**Bin packing et Cutting stock**

Professeur : M. Thierry Mautour



# ÉQUIPE



**Sarah Ouhocine**



**Djillali Boutouili**

# SOMMAIRE

01

Introduction

02

l'algorithme de  
génération de colonnes

03

cutting stock  
problem (CSP)

04

Résolution du CSP

05

Amélioration de  
l'algorithme  
Branch & Price

06

Tests et Résultats

07

Conclusion

$$T_1 = \ell_1 + 273 = 273 + 60 = 333K, T_2 = t_2 + 273 = 298K$$

$$\frac{\ell_1}{\ell} = \frac{500}{426} = 1,17$$

$$\Delta_1 = 0,01$$



$$\Delta\ell = \Delta\ell_1 + \Delta\ell_0 = 1 + 0,5 = 1,5 \text{ mm}$$



$$\begin{aligned} &+ 273 \\ &+ 273 \\ &= 327K \end{aligned}$$



# 01

## INTRODUCTION



$$\begin{aligned} e &= f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \rightarrow \\ f &= gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2 \quad dh(x) = bc \\ g &= x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h) \quad (x)^2 = ab \\ h &= ef^2 - (x)^2 + (3)^2(f)^3 + x(4x) \end{aligned}$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

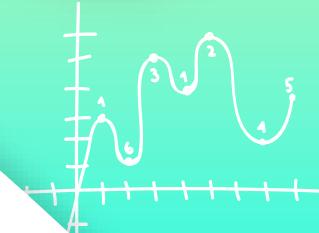
$$x^3 \div (x)(x)^2 2x \quad 2s+4x$$

$$x^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)^3}$$

$$x^2 + ab(s)^3 \quad - (x)(s)^1$$

$$x^4(OK)^3$$

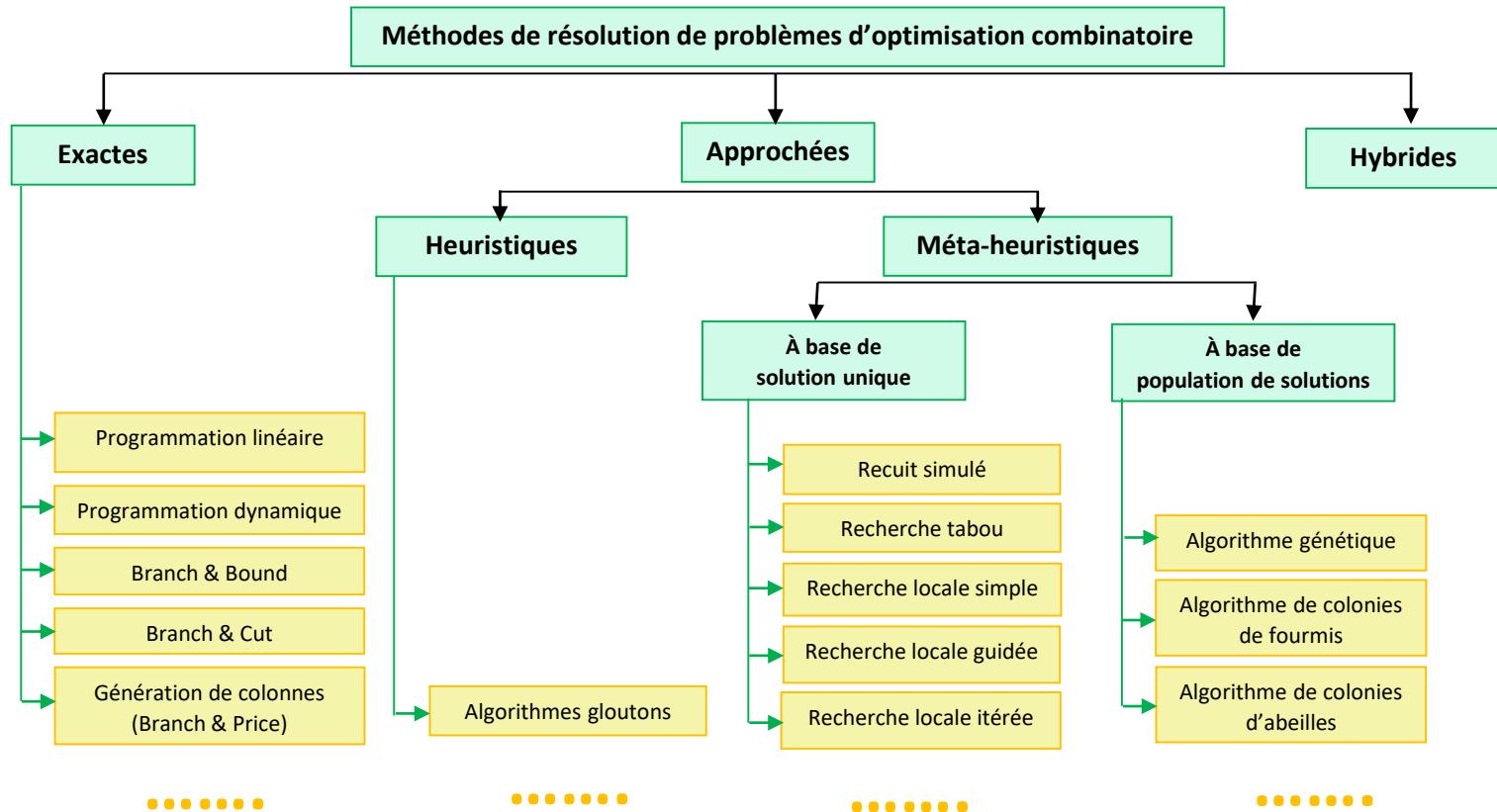


$$(x)^2 = ab$$

$$(x) = bc$$

## ✓ Optimisation Combinatoire

Trouver la meilleure solution parmi un nombre fini mais souvent très grand de solutions.





Article de

## François VanderBeck

- Professeur en Recherche Opérationnelle à l'Université de Bordeaux .
- Ses recherches portent sur les approches de la programmation en nombres entiers à l'optimisation combinatoire .
- Article publié le 19 juillet 1999.

# But de l'article

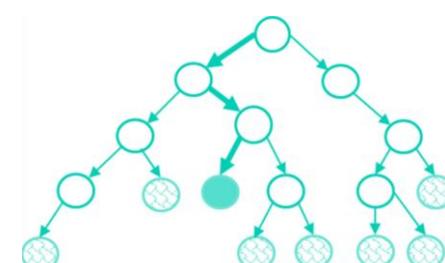
## Techniques

- 1. Variable fixing
- 2. Cutting planes
- 3. Early branching
- 4. Rounding heuristics
- 5. Different initialisations
- 6. Early terminations



# Branch & Bound

- Méthode exacte pour la résolution d'un problème d'optimisation
- À recherche arborescente
- Utilise des relaxations du problème original
  - Relaxation continue
  - Relaxation lagrangienne



# 02

## L'ALGORITHME DE GÉNÉRATION DE COLONNES



$$\begin{aligned}
 e &= f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \\
 f &= gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2 \\
 g &= x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h) \\
 h &= ef^2 - (x)^2 + (3)^2(f)^3 + x(4x)
 \end{aligned}$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

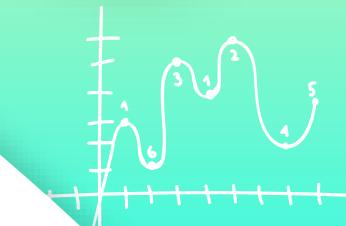
$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$e^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)^3}$$

$$x^2 + ab(s)^3 - (x)(s)^1$$



$$(x)^2 = ab$$

$$(x) = bc$$



5



$$F_1 \neq F_2$$



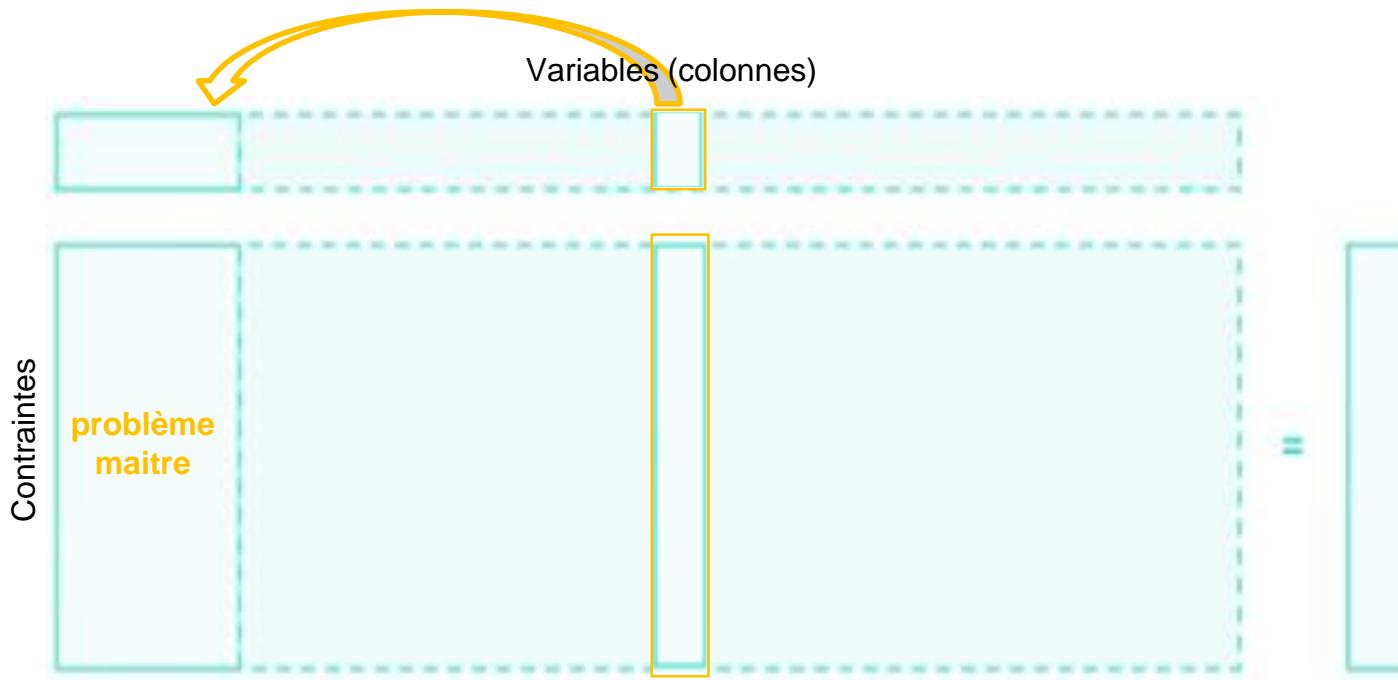
$$F_1$$

$$F_2$$

# L'Algorithme de génération de colonnes

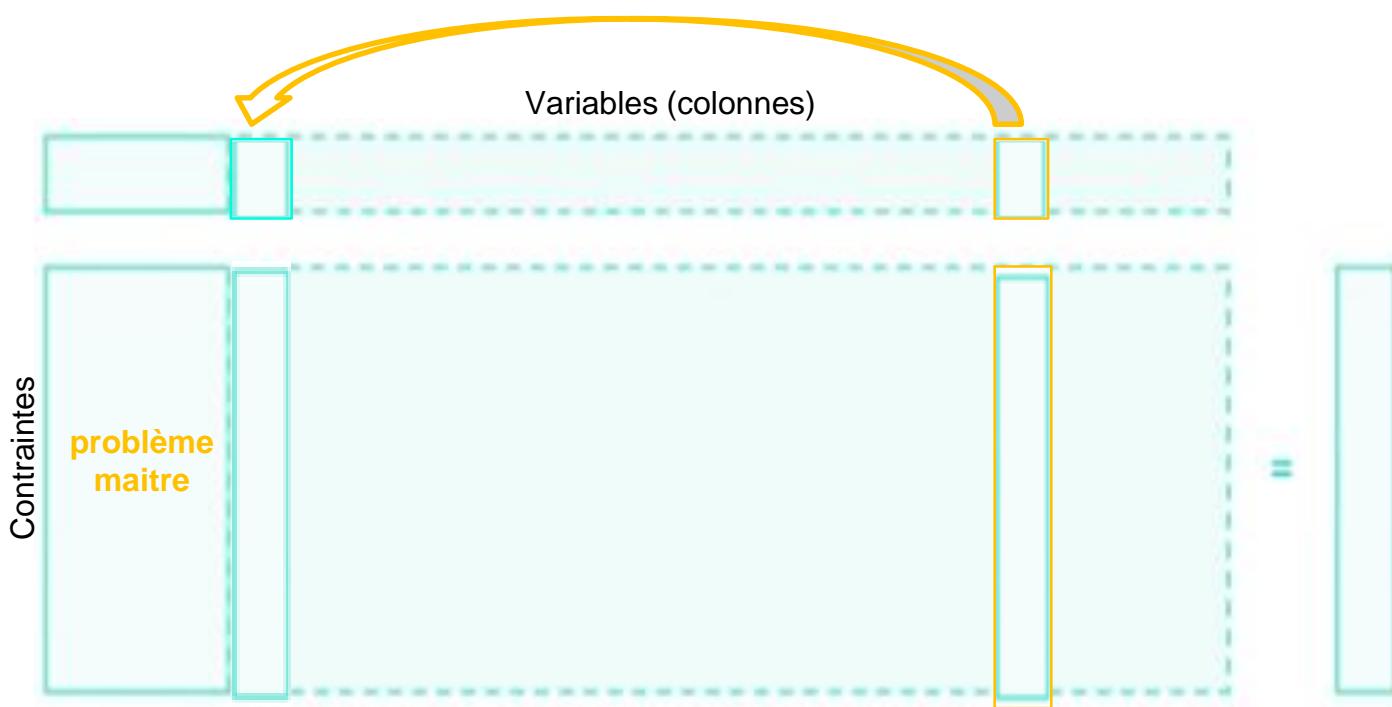
Résoudre un problème d'optimisation linéaire de grande taille (avec beaucoup de variables)

- ✓ Résoudre avec un sous ensemble de variables (problème maître)



# L'Algorithme de génération de colonnes

- ✓ Méthode exacte pour résoudre un problème d'optimisation linéaire de grande taille (avec beaucoup de variables)
  - ✓ Résoudre avec un sous ensemble de variables (problème maître)





7

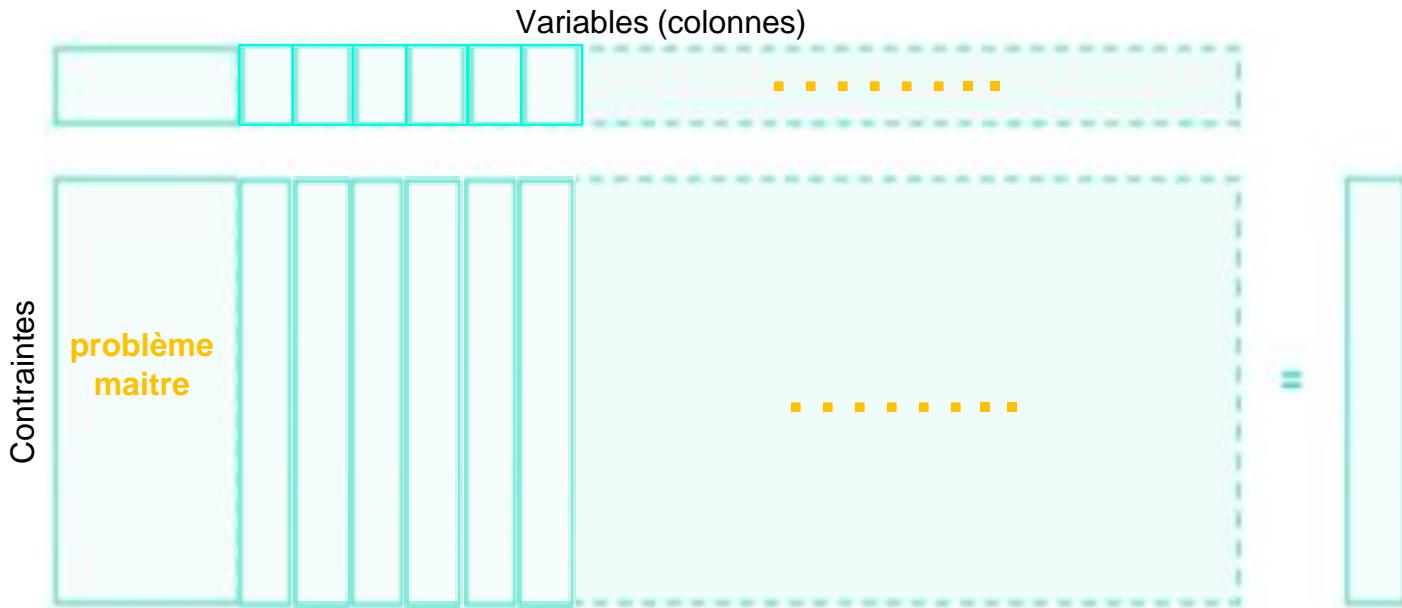


$$F_1 \neq F_2$$



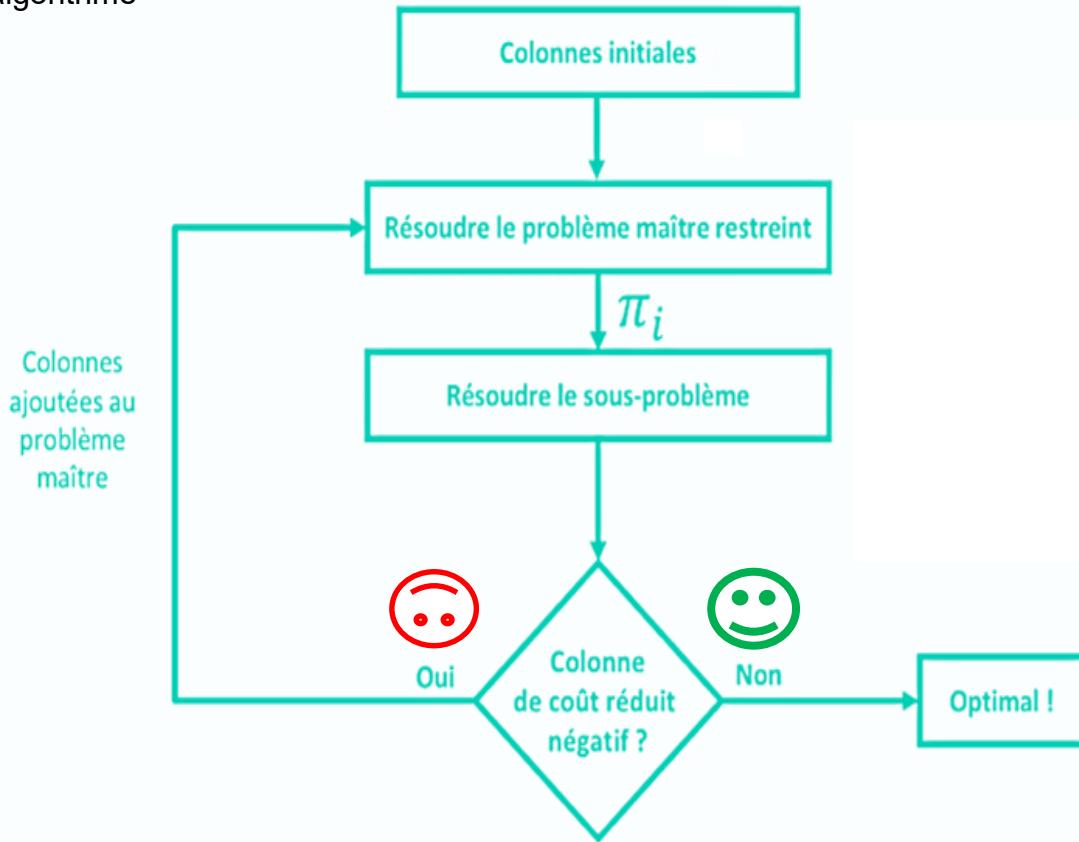
# L'Algorithme de génération de colonnes

- ✓ Résoudre un problème d'optimisation linéaire de grande taille (avec beaucoup de variables)
  - ✓ Résoudre avec un sous ensemble de variables (problème maître)



# L'Algorithme de génération de colonnes

- ✓ Structure de l'algorithme



# L'Algorithme de génération de colonnes

## Branch & Price

- Génération de colonnes + PLNE

Comment obtenir donc des solutions entières ?

Branch & Bound

Résoudre  
la relaxation linéaire  
à chaque nœud

Branch & Price

Génération de colonnes  
pour résoudre  
la relaxation linéaire  
à chaque nœud

# L'Algorithme de génération de colonnes

Quand utiliser le Branch & Price ?

- Tournée de véhicules
- Planning aérien (construction de routes pour les avions)
- Rotations horaires
- Ordonnancement de tâches
- Découpes et d'emballages (**Cutting Stock Problem & Bin packing**)

# 03

## CUTTING STOCK PROBLEM



$$\begin{aligned} e &= f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \rightarrow \\ f &= gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2 \quad dh(x) = bc \\ g &= x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h) \quad (x)^2 = ab \\ h &= ef^2 - (x)^2 + (3)^2(f)^3 + x(4x) \end{aligned}$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$x^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)^3}$$

$$x^2 + ab(s)^3 \quad - (x)(s)^1$$

$$x^4(OK)^3$$



$$(x)^2 = ab$$

$$(x) = bc$$

## Cutting Stocks Problem (CSP)



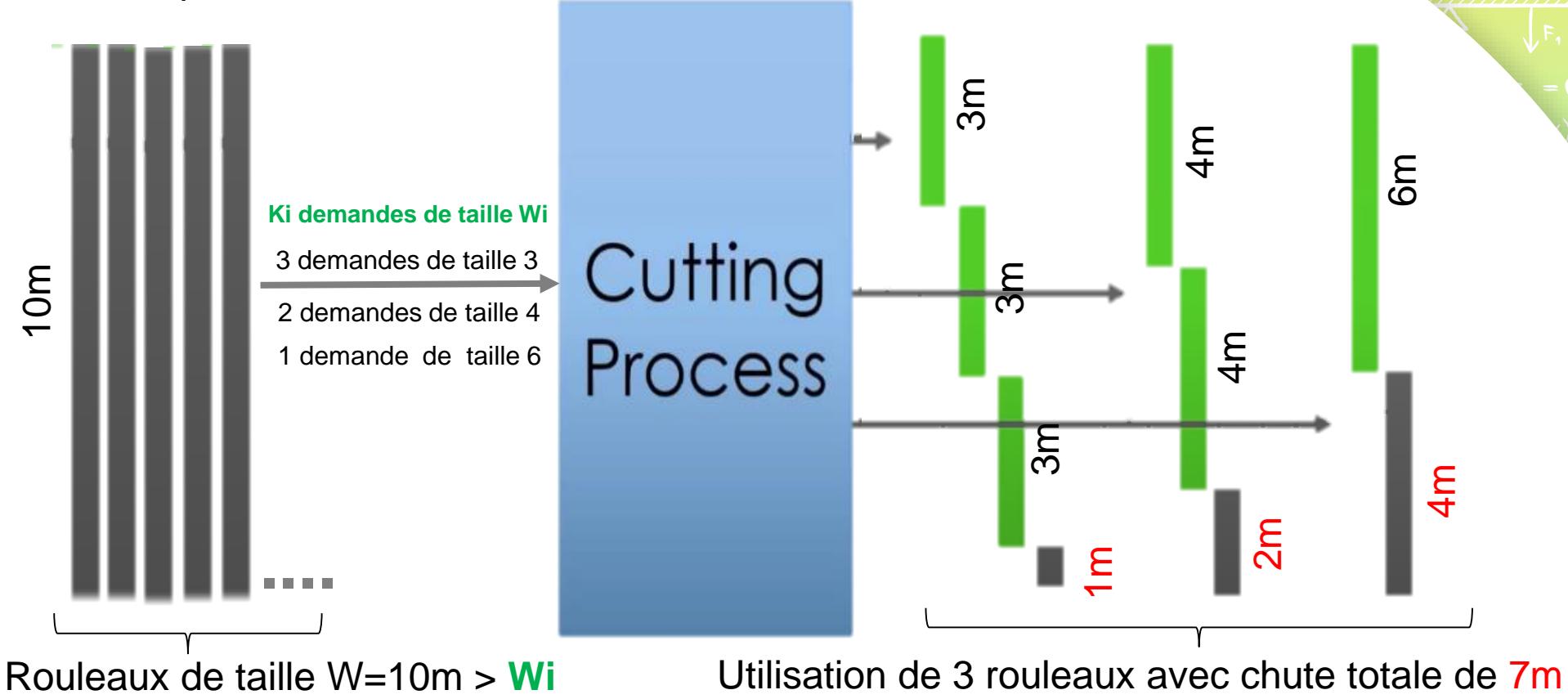
- Un problème commercial qui se pose dans de nombreuses industries : le textile, le cuir, le papier, le bois, le verre, la tôle, etc...
- Le but est de découper les commandes des clients d'une taille donnée dans des larges rouleaux en minimisant la chute totale lors du découpage
- La chute totale : la perte correspondant aux parties inutilisées des rouleaux entamés.



# Cutting Stock Problem (CSP)



Exemple illustratif



# Cutting Stock Problem (CSP)



- Minimiser la chute totale est équivalente à minimiser le nombre total de rouleaux utilisés.

$$\sum \text{Chutes} = W \times \text{nombre de rouleaux utilisé} - \sum \text{Demandes}$$

*↑ Constante*                                   *↑ Constante*

$$\sum \text{Chutes} = \text{nombre de rouleaux utilisé}$$



- Il existe un cas particulier du CSP qui est le Bin Packing (BP).

- Bin Packing** : CSP où les demandes se font par unité ( $K_i = 1$ ).

$$\text{CSP } (K_i, W_i) = (3, 3), (2, 4), (1, 6).$$

$$\text{BP } (K_i, W_i) = (1, 3), (1, 4), (1, 6).$$

# Cutting Stock Problem (CSP)



- Modélisation classique

$$\text{MINIMISER } \sum_{k=1}^P y_k$$

$$\forall i \in 1..n, \sum_{k=1}^P x_{ik} = k_i$$

$$\forall k \in 1..P, \sum_{i=1}^n w_i \cdot x_{ik} \leq W \cdot y_k$$

$$\forall i \in 1..n, \forall k \in 1..P, x_{ik} \in \mathbb{N}, y_k \in \{0, 1\}$$

- Signification des variables

$(k_i, w_i)$  :  $k_i$  demandes de taille  $w_i$

$n$  : nombre de demandes de taille différente

$P$  : nombre de rouleaux au total

$W$  : taille du rouleau (tous les rouleaux sont de taille identique et elle est supérieure à la taille de chaque demande  $w_i$ )

$x_{ik}$  : nombre de demandes de type  $i$  (taille  $w_i$ ) découpées dans le même rouleau ( $k^e$  rouleau)

$y_k$  : nombre de rouleaux utilisé

- Signification de la fonction objectif

Minimiser le nombre de rouleaux utilisé

$$\begin{cases} Y_k = 1 \text{ si le } k^e \text{ rouleau est utilisé} \\ Y_k = 0 \text{ si le } k^e \text{ rouleau n'est pas utilisé} \end{cases}$$

- Signification des contraintes

→ **Contrainte 1** : Toutes les demandes des clients doivent être satisfaites.

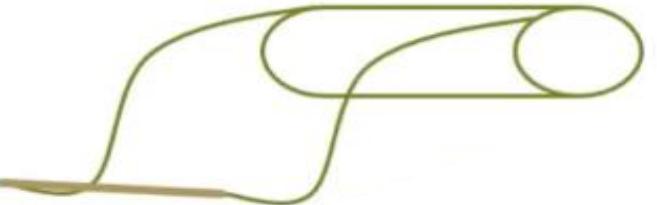
→ **Contrainte 2** : Respecter la taille max du rouleau  $W$

$$\begin{cases} Y_k = 1 \text{ si au moins un } x_{ik} > 0 \\ Y_k = 0 \text{ sinon} \end{cases}$$

# Cutting Stock Problem (CSP)



- Un tel découpage est une activité récurrente qui nécessite une planification appropriée.
- De nombreuses méthodes de résolutions ont été proposées.



# 04

## RÉSOLUTION DU CSP



$$e = f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \rightarrow$$

$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2$$

$$g = x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h)$$

$$h = efg^2 - (x)^2 + (3)^2(f)^3 + x(4x)$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

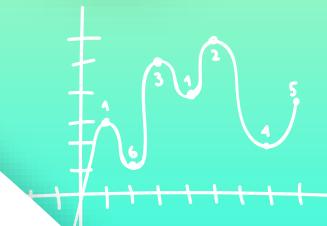
$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$c^2(h)$$

$$\left. ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)} \right\}$$

$$x^2 + ab(s)^3$$

$$- (x)(s)^1$$



$$(x)^2 = ab$$

$$(x) = bc$$

$$F_1 \neq F_2$$

16

# Résolution du CSP avec Méthodes Approchées

## 1. First-Fit Decreasing (FFD)

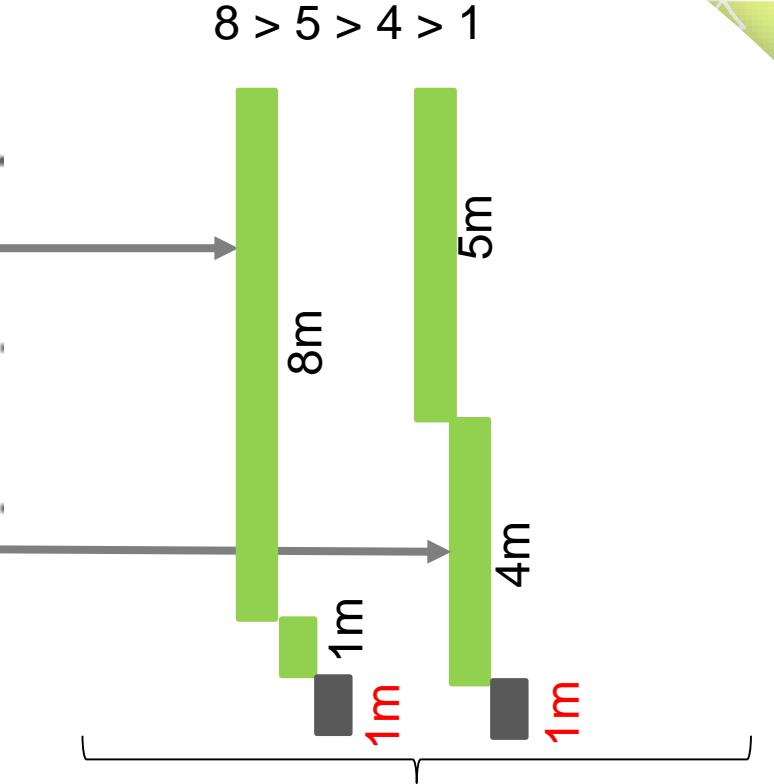
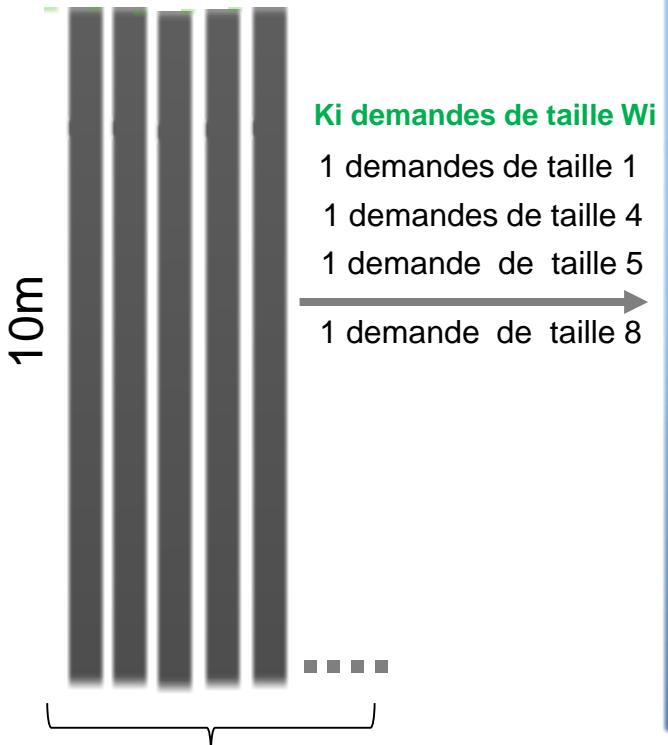
- Trier les commandes par taille décroissante.
- Traiter successivement les commandes dans cet ordre t.q chaque commande soit affectée au premier rouleau qui peut l'accueillir.



FFD



## Exemple illustratif



$F_1 \neq F_2$

18

## Résolution du CSP avec Méthodes approchées

### 2. Best-Fit Decreasing (BFD)

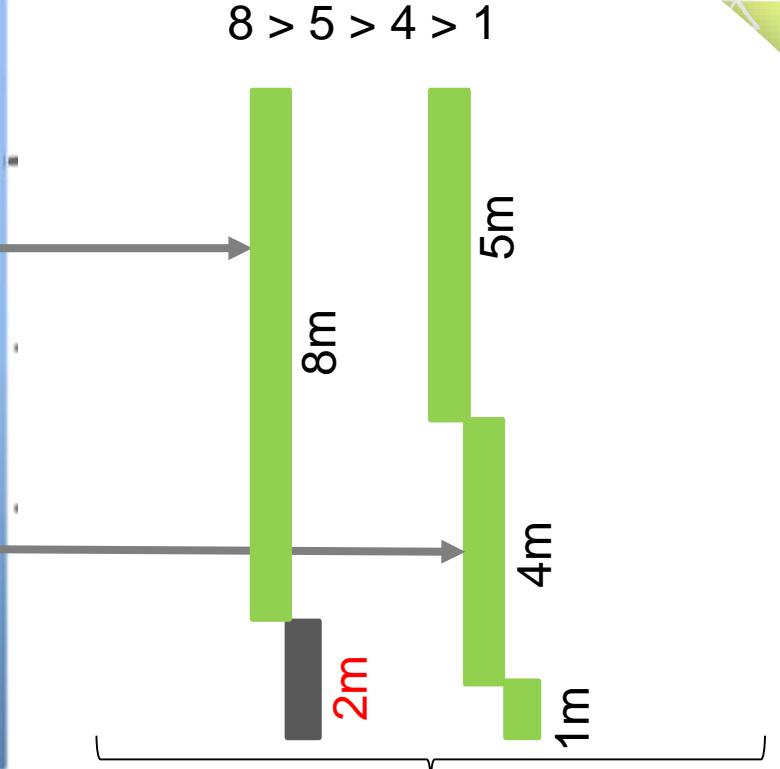
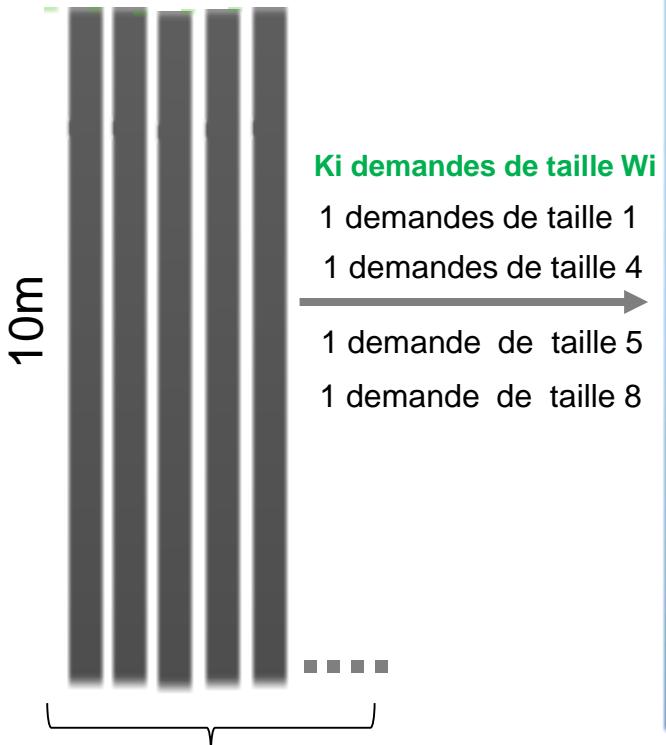
- Trier les commandes par taille décroissants.
- Traiter successivement les commandes dans cet ordre t.q chaque commande soit affectée au premier rouleau **le plus serré** qui peut l'accueillir.
- Algorithme d'approximation légèrement meilleur que le FFD (chutes plus longues sur moins de rouleaux).



# BFD



## Exemple illustratif



Rouleaux de taille W=10m > Wi



## Résolution du CSP avec Méthodes approchées

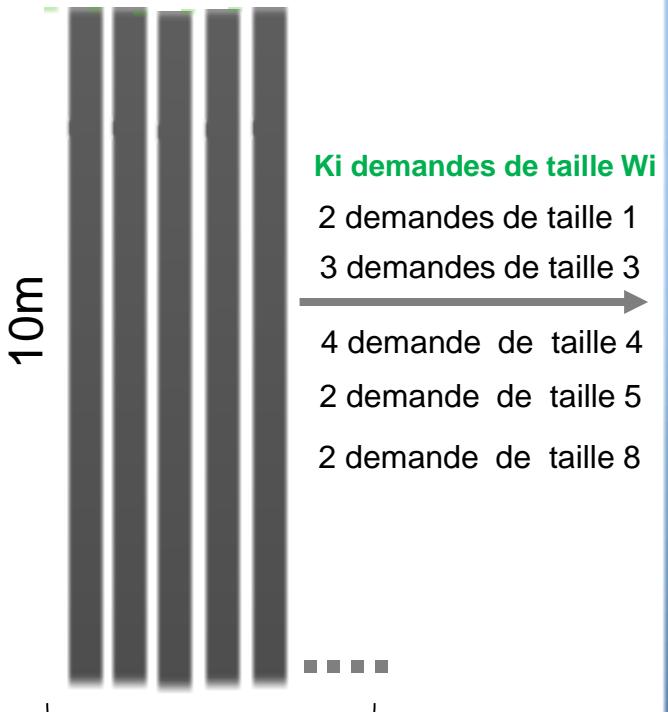
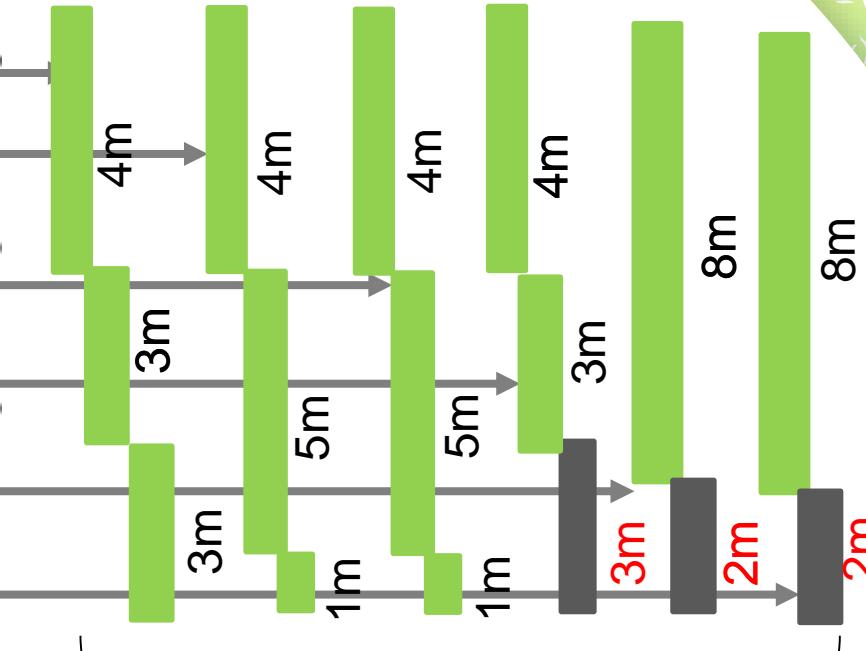
### 3. Fill Bin heuristic (FBh)

- Se concentre sur l'utilisation entière des rouleaux en découpant les commandes qui peuvent rentrer.
- L'heuristique peut engendrer beaucoup de chutes à la fin de la procédure de découpage.





## Exemple illustratif

Rouleaux de taille  $W=10m > \text{Wi}$ Chute totale de  $7m$  sur 3 rouleaux

## ❖ Les méthodes Approchées



ne garantissent pas toujours



La solution optimale

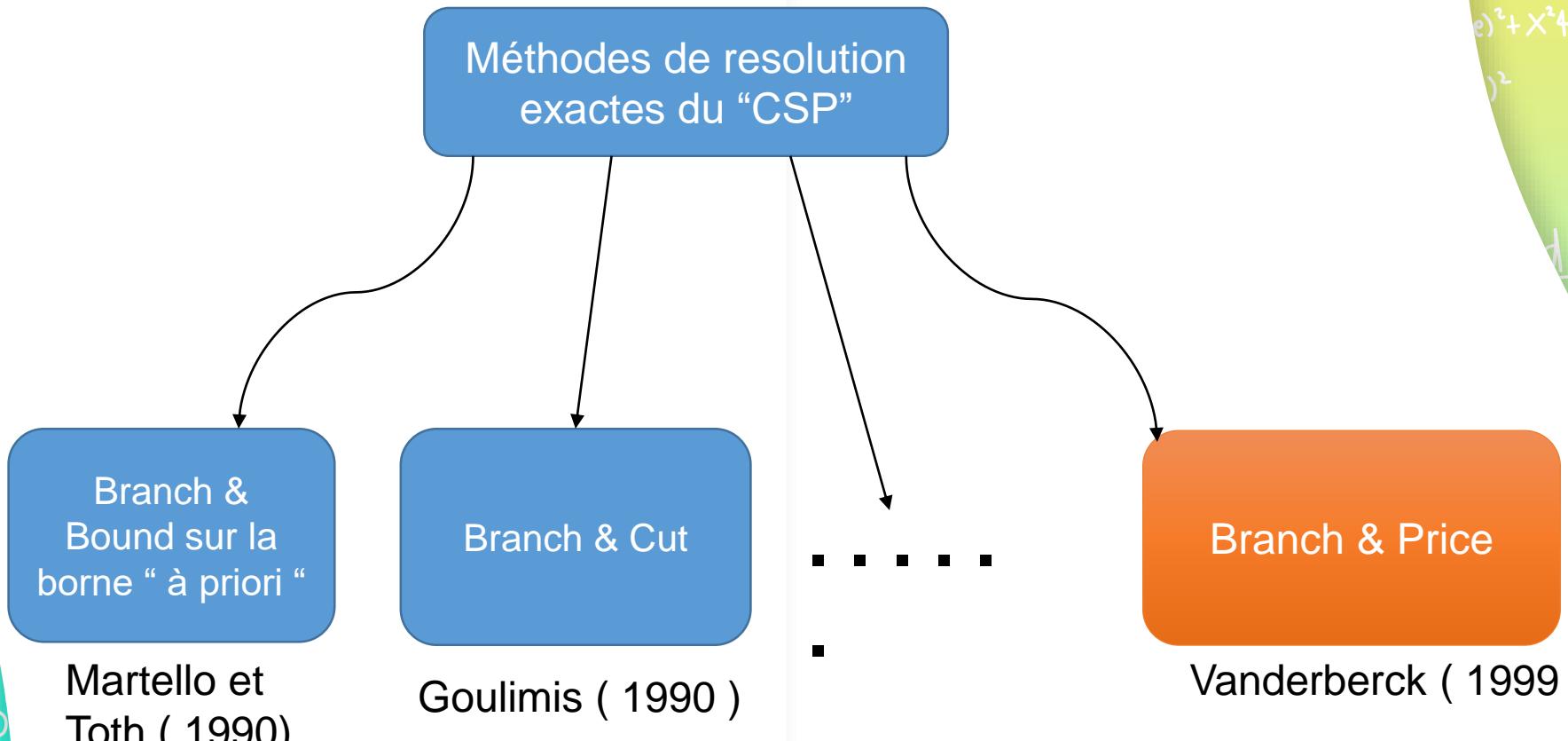


Solutions ?



## ❖ Les méthodes Exactes

## Résolution par Méthodes Exactes

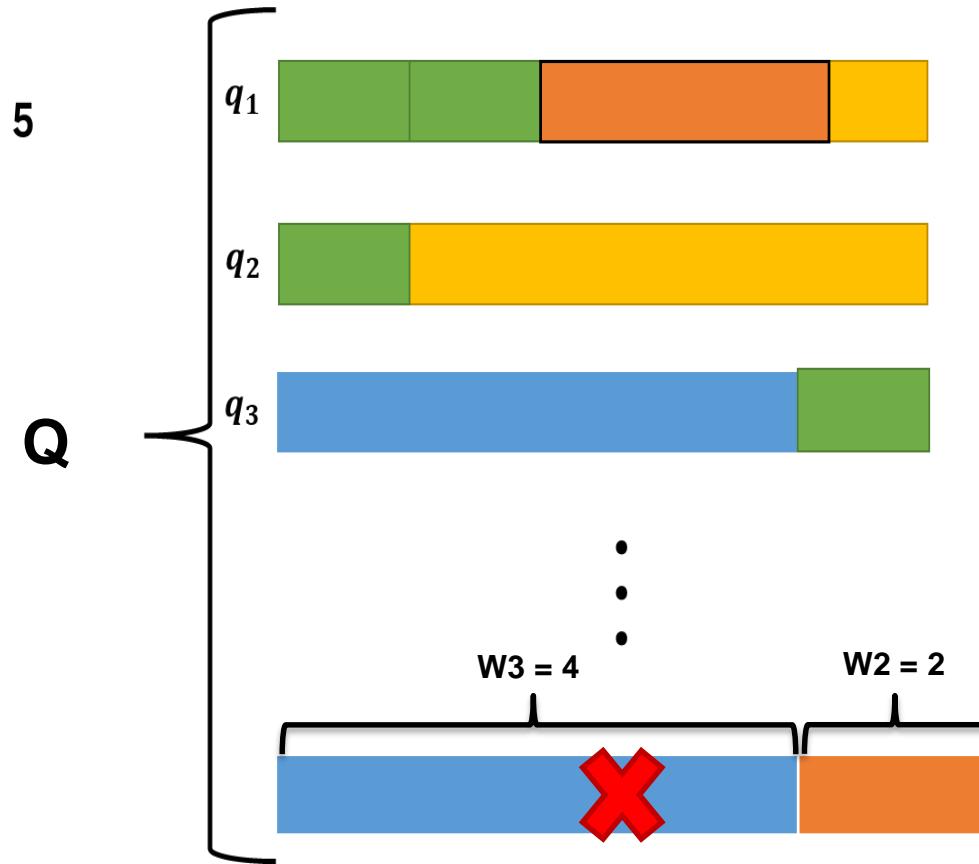
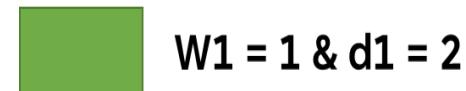


# Modélisation du CSP Branch And Price

- $Q$  ensemble des combinaisons **faisables** du même rouleau.
- $\gamma_q$  le nombre de fois la combinaison "q" est utilisée.
- $d_i$  étant toujours les demandes de l'item "i"
- $q_i$  étant le nombre d'occurrences de l'item "i" dans la combinaison q

$$\begin{aligned}
 Z^M &= \min \quad \sum_{q \in Q} \lambda_q \\
 &\text{s.t.} \\
 &\sum_{q \in Q} q_i \lambda_q \geq d_i \\
 &\lambda_q \in \mathbb{N}
 \end{aligned}$$

Quoi de mieux qu'un exemple pour éclaircir tout ça



$$\begin{aligned} u t + \frac{1}{2} a t^2 \\ v = u + a \\ w = F \cdot \gamma \end{aligned}$$

# Formulation du sous-problème

$$\max \left\{ \sum_i w_i x_i : \sum_i w_i x_i \leq W, x_i \leq d_i \text{ for } i = 1, \dots, n, x \in \mathbb{N}^n \right\}$$

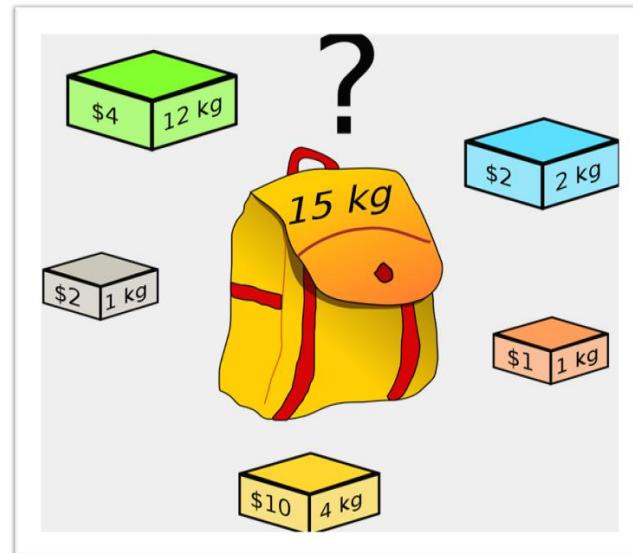


**Problème de sac à dos**

- $\Pi$  étant le vecteur des variables dual associé à la contrainte des demandes.

Résolvable de façon polynomiale par :

- programmation dynamique.
- Branch & bound.



ut +  $\frac{1}{2}$  at  
 $v = u + c$   
 $w = F \cdot \gamma_8$



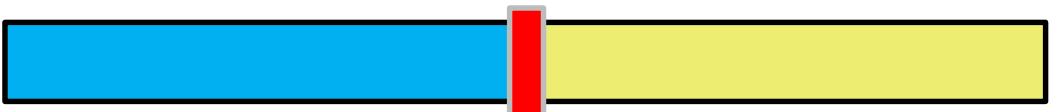
27



$$F_1 \neq F_2$$

 $F_1$  $F_2$ 

# Lower Bounds

 $X^*$ 

- Les chercheurs s'interessent toujours à borner les problèmes d'optimisation combinatoire.
- $L^1$  La borne trivial :

$$L^1 = \lceil Z_{LP}^F \rceil$$

$$Z_{LP}^F = \frac{\sum_{i=1}^n w_i d_i}{W}$$

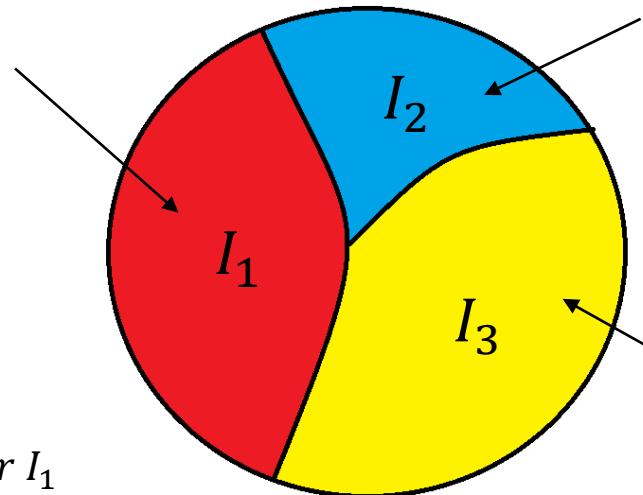
# Lower Bounds

- $L^2$  La borne “à priori” :

$$0 \leq \alpha \leq \frac{W}{2}$$



$$\frac{W}{2}$$



$$\frac{W}{2}$$



$$\alpha$$

$$\frac{W}{2}$$

: Partie non utilisée

•  $D^1$  : # demandes sur  $I_1$

: Partie utilisée

•  $D^2$  : # demandes sur  $I_2$

$$L(\alpha) = D^1 + D^2 + \max \left\{ 0, \left\lceil \frac{\sum_{i \in (I^2 \cup I^3)} w_i d_i}{W} \right\rceil - D^2 \right\}$$

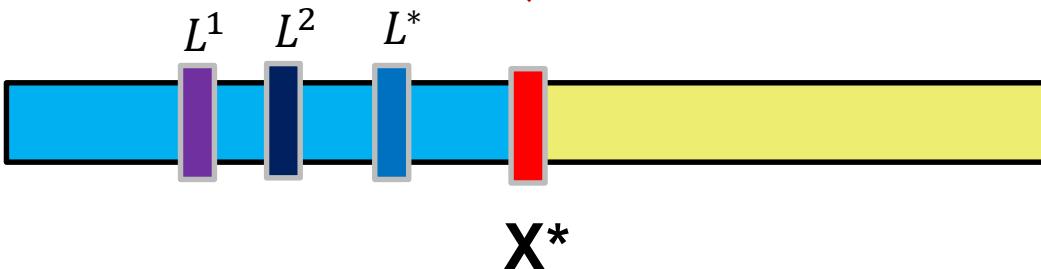
$$L^2 = \max(L(\alpha))$$

# Lower Bounds

$$L^* = \lceil Z_{LP}^M \rceil$$

Comme  $M_{LP}$  est la dualization de lagrange de  $F_{LP}$  :

$$Z_{LP}^F \leq Z_{LP}^M$$



# 06

## AMÉLIORATION DE L'ALGORITHME BRANCH & PRICE



$$\begin{aligned} e &= f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \\ f &= gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2 \\ g &= x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h) \\ h &= ef^2 - (x)^2 + (3)^2(f)^3 + x(4x) \end{aligned}$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

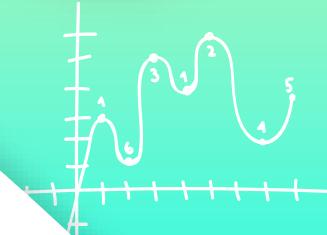
$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$x^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)_3}$$

$$x^2 + ab(s)^3 - (x)(s)^1$$

$$x^4(OK)^3$$



$$(x)^2 = ab$$

$$(x) = bc$$



$$\begin{aligned}T_1 &= t_1 + 273 = 273 + 60 = \\&= 333K, T_2 = t_2 + 273 = 298K\end{aligned}$$

$$\frac{l_1}{l} = \frac{500}{426} = 1,17$$

$$\Delta_1 = 0,01$$



$$\begin{aligned}\Delta T &= \Delta T_{in} + \Delta T_o = 1+0,5 = 1,5K \\&\quad + 273 \\&\quad + 273 \\&= 327K\end{aligned}$$



01

## Initialisations

03

## Variable Fixing

02

## Cutting planes

04

## Rounding Heuristic

# 1.1 Initialisations

- Initialiser  $\mathbf{W}$  avec la longueur maximale pratique  $\mathbf{W}'$  :



$$W = 7$$



$$W_1 = 2$$



$$W_2 = 4$$



$$W_3 = 6$$



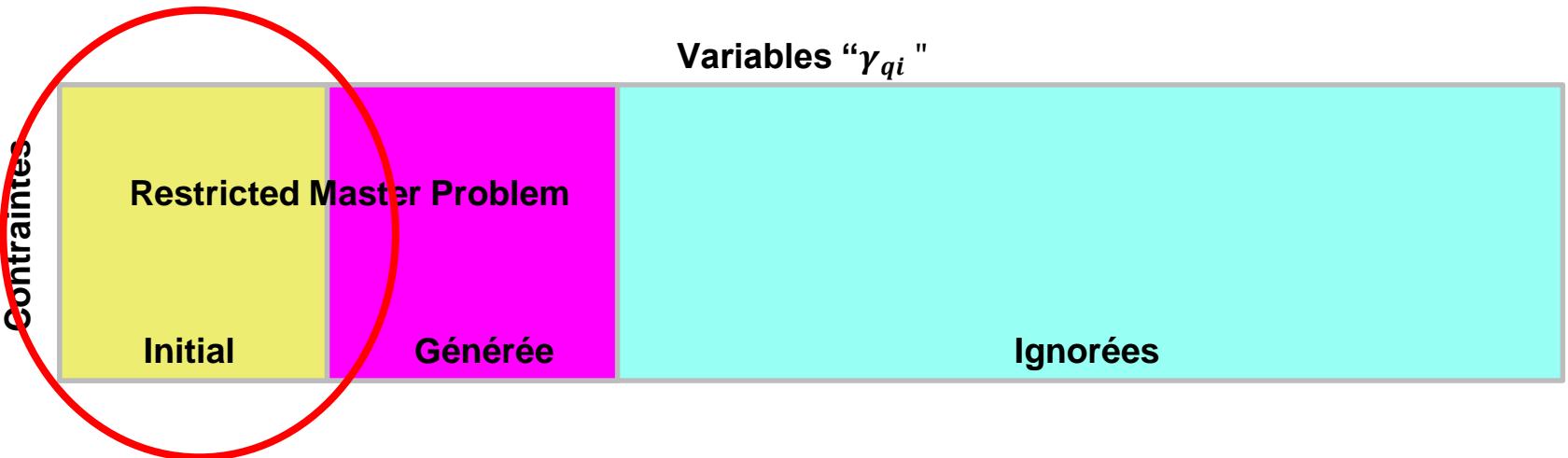
$$W' = 6$$



$$W := W'$$

## 1.2 Initialisations

- Initialisation du programme maître



Les variables choisies en entrée doivent contenir une solution réalisable du “ RMP “

# 1.3 Initialisations

- Initialisation **Matrice Unité**

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$q_1 = (1, 0, 0, 0)$$



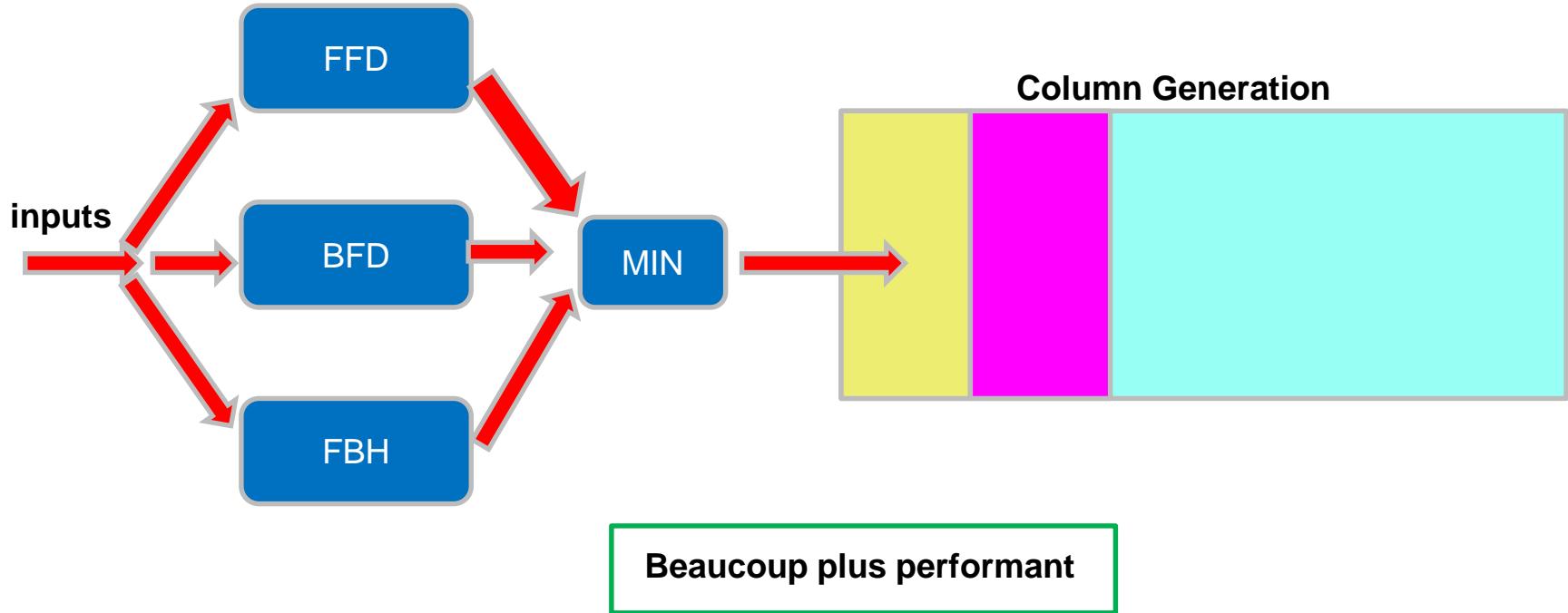
$$q_2 = (0, 1, 0, 0)$$



Très lente convergence

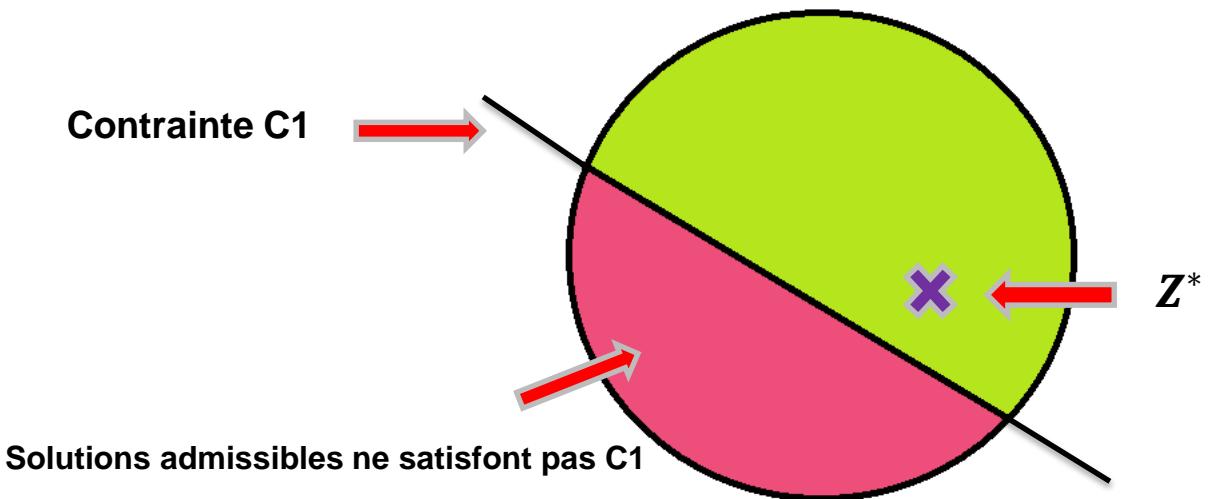
## 1.4 Initialisations

- Initialisation **par le meilleur résultat des 3 heuristiques**



## 2.1 Cutting Planes

- Objectif : Renforcer le programme maître et rendre la convergence plus rapide



## 2.2 Cutting Planes

- Feasibility cuts :

$$S \subseteq \{1, \dots, n\}$$

$$\sum_{q \in Q: q \cap S \neq \emptyset} \lambda_q \geq \left\lceil \frac{\sum_{i \in S} w_i d_i}{W} \right\rceil$$

- Coupe 2

$$S = \{ i \}$$

$$\sum_{q \in Q: q_i > 0} \lambda_q \geq \left\lceil \frac{d_i}{q_i^{\max}} \right\rceil$$



Nombre maximal de copies de  $i$  pouvant être mis dans un rouleau de taille  $W$

$$q_i^{\max} \leq 3$$



37



$$F_1 \neq F_2$$

 $F_1$  $F_2$  $=$  $\neq$ 

## 2.3 Cutting Planes

- Lower Bound Cut :

$$\sum_q \lambda_q \geq L$$

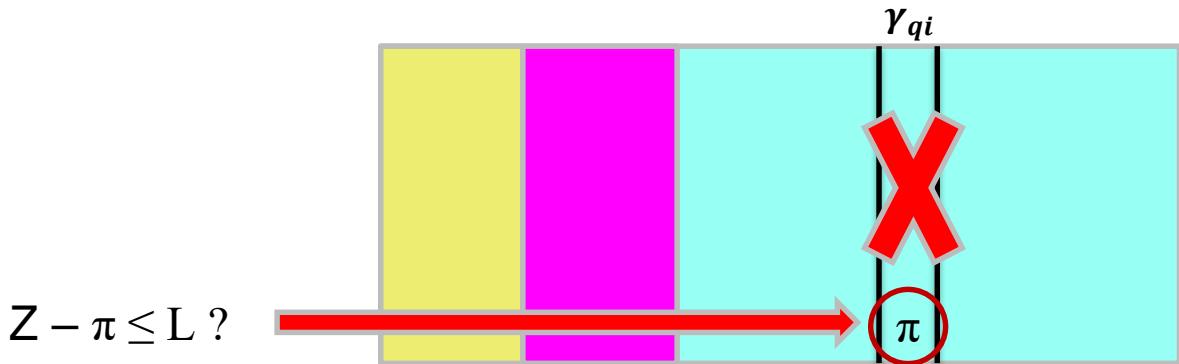
Coupe Triviale

## 3.1 Variable Fixing

- Fixer ou borner la valeur d'une variable
- Se base principalement sur les conditions d'optimilité et de faisabilité

Reduced cost fixing :

- Technique qui utilise le coût réduit de la variable pour la fixer.



## 3.2 Variable Fixing

- Technique pas performante sur un algorithme de génération de colonnes.
- Comment faire pour appliquer cette technique sur plusieurs colonnes au même temps?

### Les Variables Duales

- Appliquer cette technique sur le sous-problème en fixant des éléments du problème de sac à dos à Zéro.
- Tests coûteux, et pas beaucoup d'intérêt en performances.



40



$$F_1 \neq F_2$$



$$F_1$$

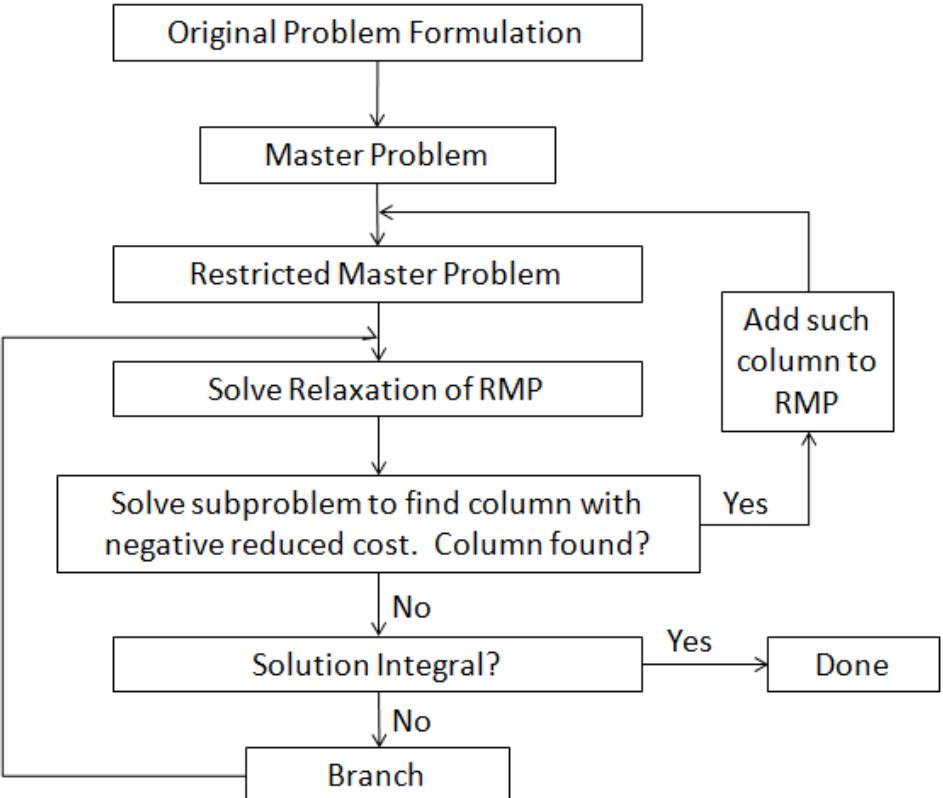


$$F_1 = 0$$



## 4.1 Rounding Heuristic

- Schéma de l'algorithme du “ Branch & Price “



## 4.2 Rounding Heuristic

- La solution optimale du “PL” n'est quasiment jamais entière.
- Pas facile de retrouver la solution entière à partir de la solution du problème relâché, dans **la majorité des cas ...impossible !!!**
- Processus très très couteux.

La solution ? Rounding Heuristic

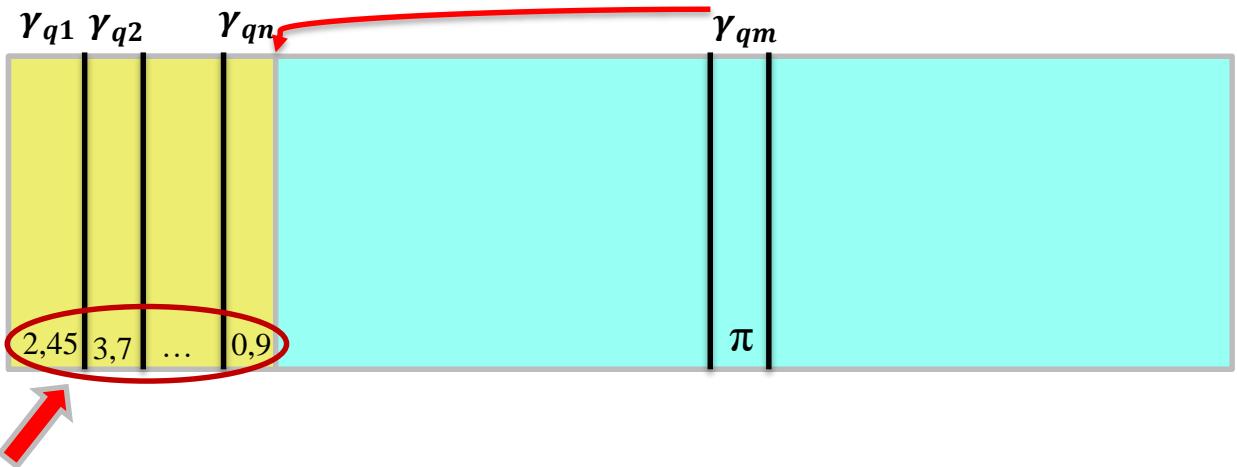
## 4.3 Rounding Heuristic

- Ne garantit pas toujours la valeur optimale mais le fait à **99%** des cas !
- depth first search heuristic
- Construire une solution entière partielle en arrondissant la solution du “RMP” vers le bas à chaque nœud.
- Mise à jour des variables restantes du problème maître.
- Très performante quand combinée avec les méthodes de coupes.

$$q_i^{max} \leq 3$$

## 4.4 Rounding Heuristic

- Illustrons tout ça :



$$d_i = \max\{0, d_i - \sum_q q_i \lfloor \lambda_q \rfloor\}$$

- $d_i$  plus petit et donc PM et sous-problème plus petit  $\rightarrow q_i^{max} \leq 3$

# 06

## Tests Et Résultats



$$\begin{aligned} e &= f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \quad \text{---} \\ f &= gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2 \quad dh(x) = bc \\ g &= x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h) \quad (x)^2 = ab \\ h &= ef^2 - (x)^2 + (3)^2(f)^3 + x(4x) \end{aligned}$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$x^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)^3}$$

$$x^2 + ab(s)^3 - (x)(s)^1$$



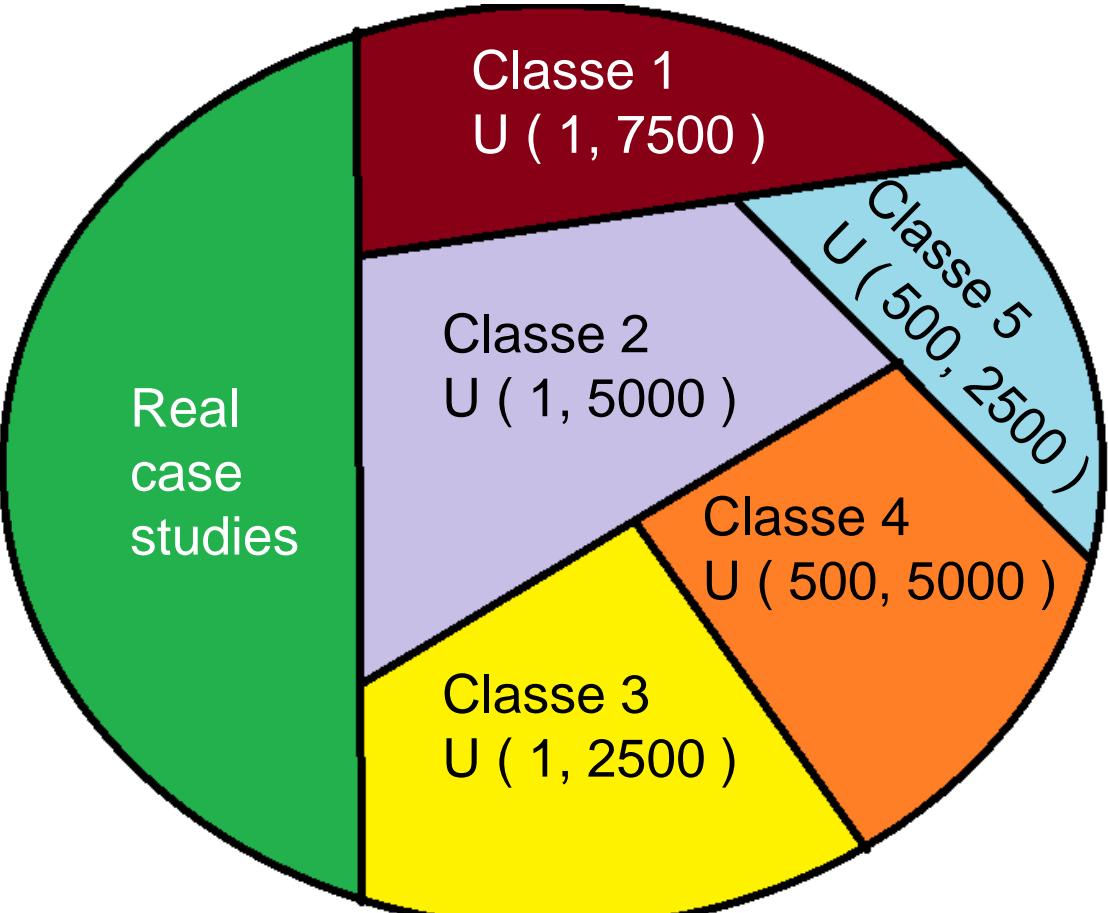
$$\begin{aligned} (x)^2 &= ab \\ (x) &= bc \end{aligned}$$

## 1.1 Environnement de test

- Langage utilisé : C
- Les programmes maîtres sont résolus sous le solveur CPLEX 3.0
- Le sous-problème par du Branch & Bound.
- Machine HP90000/752/80 avec 64MB de RAM

## 1.2 Instances de test

- $W = 10000$
- $N = 50$  items

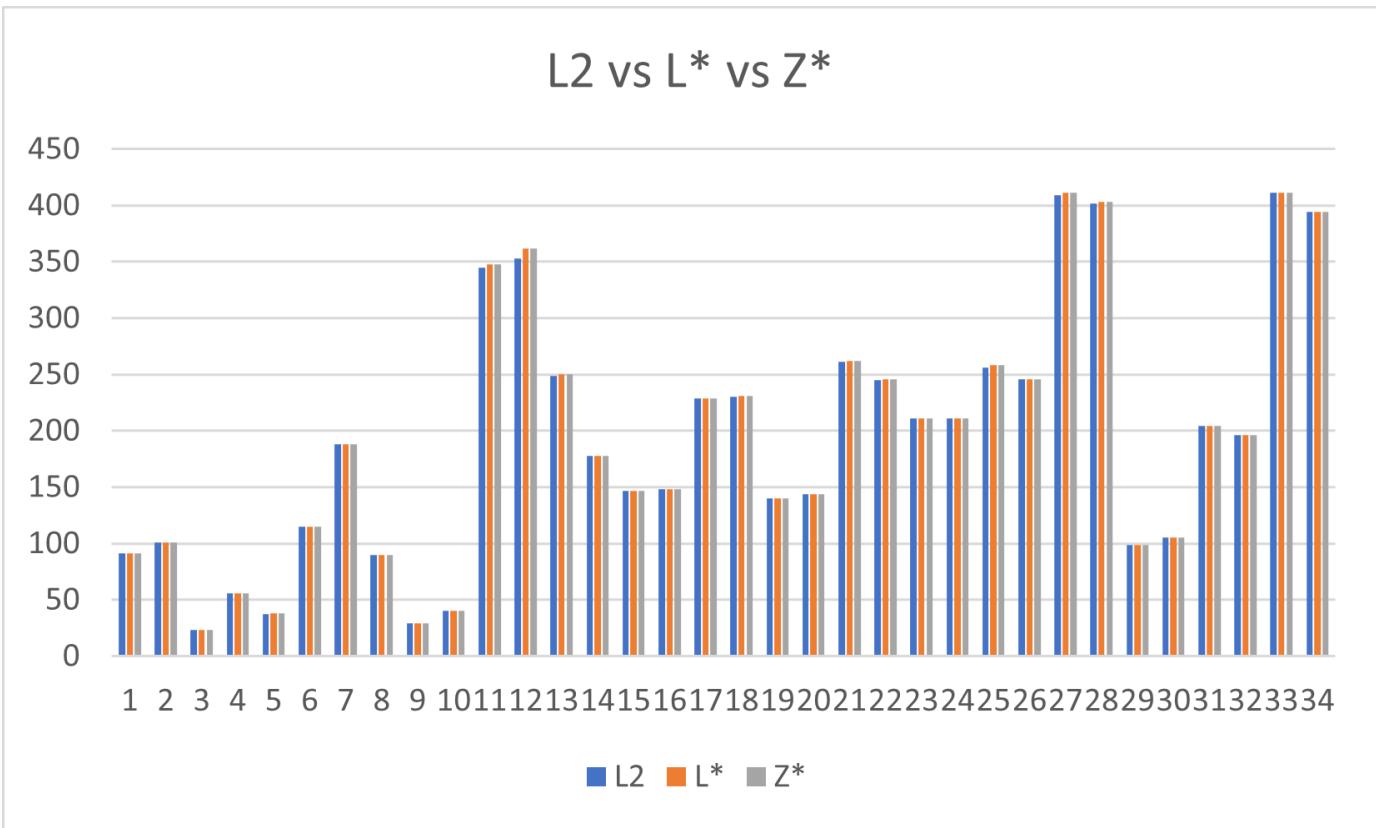


## 2.1 Résultats

- Développement d'un programme “ Benchmark ”
- Init avec résultats d'heuristiques
- Cutting planes X
- Variable fixing X
- Rounding heuristic X



## 2.2 Résultats



$L^* = Z^*$  et L2 sous-estime  $Z^*$  de 0,35% !!

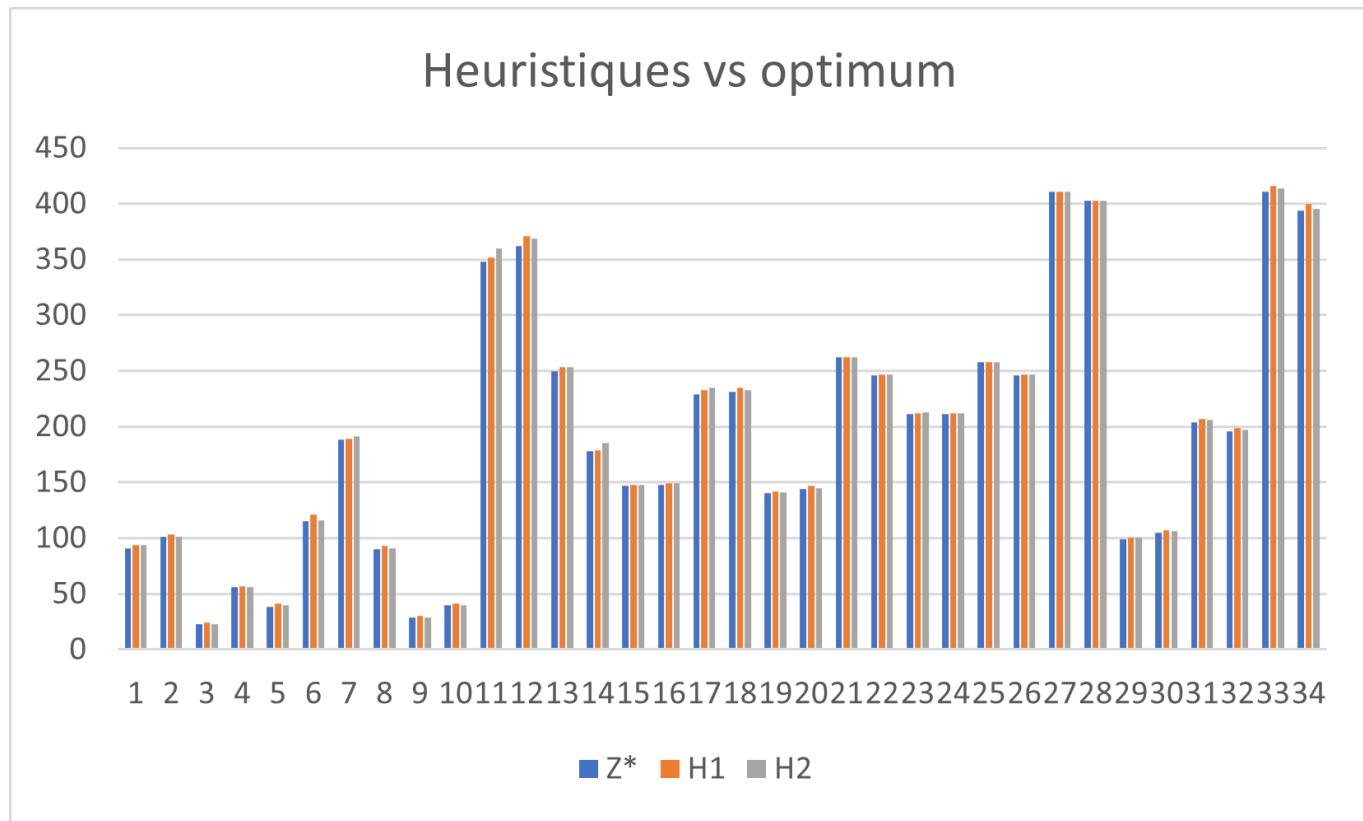


$$F_1 \neq F_2$$

$$F_1 = F_2$$

$$F_1 = F_2$$

## 2.3 Résultats

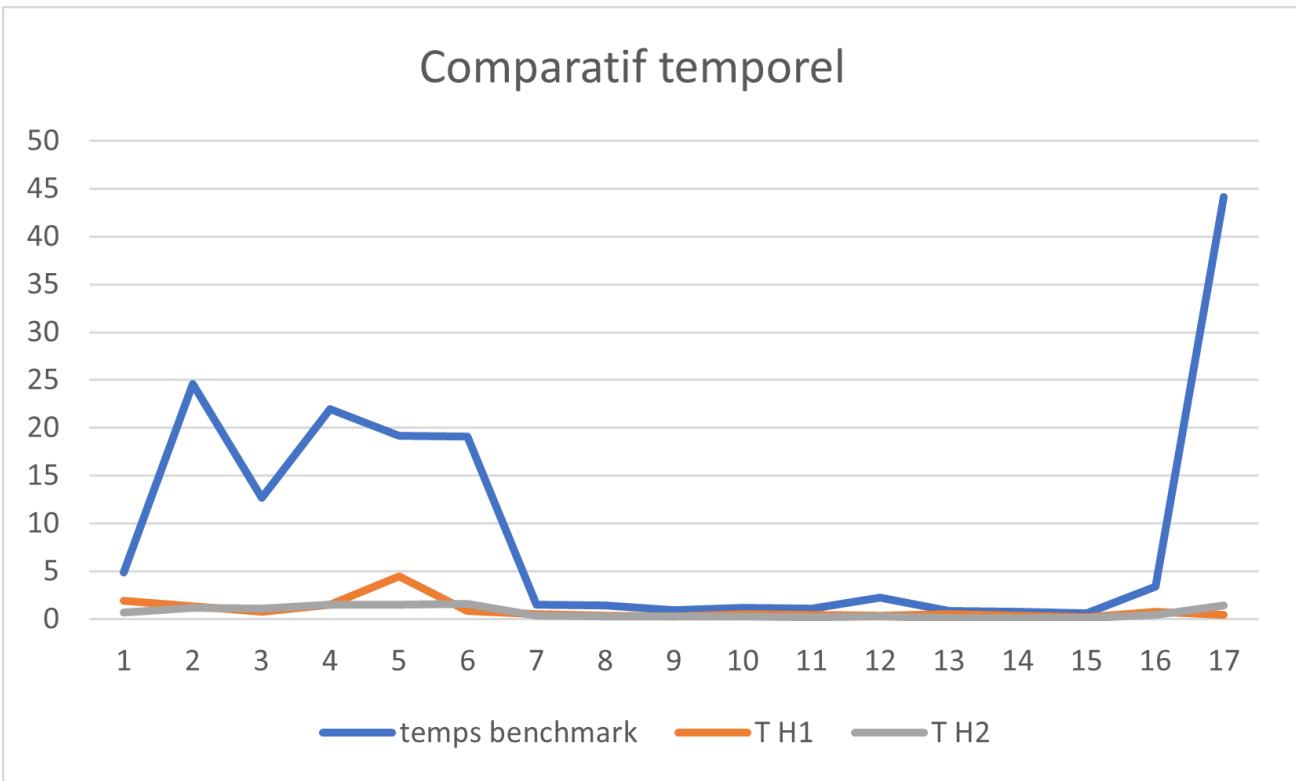


En moyenne 1.19% and 0.99% de plus que l'optimum



$$F_1 \neq F_2$$


## 2.4 Résultats



Grande différence de performance



50



$$F_1 \neq F_2$$

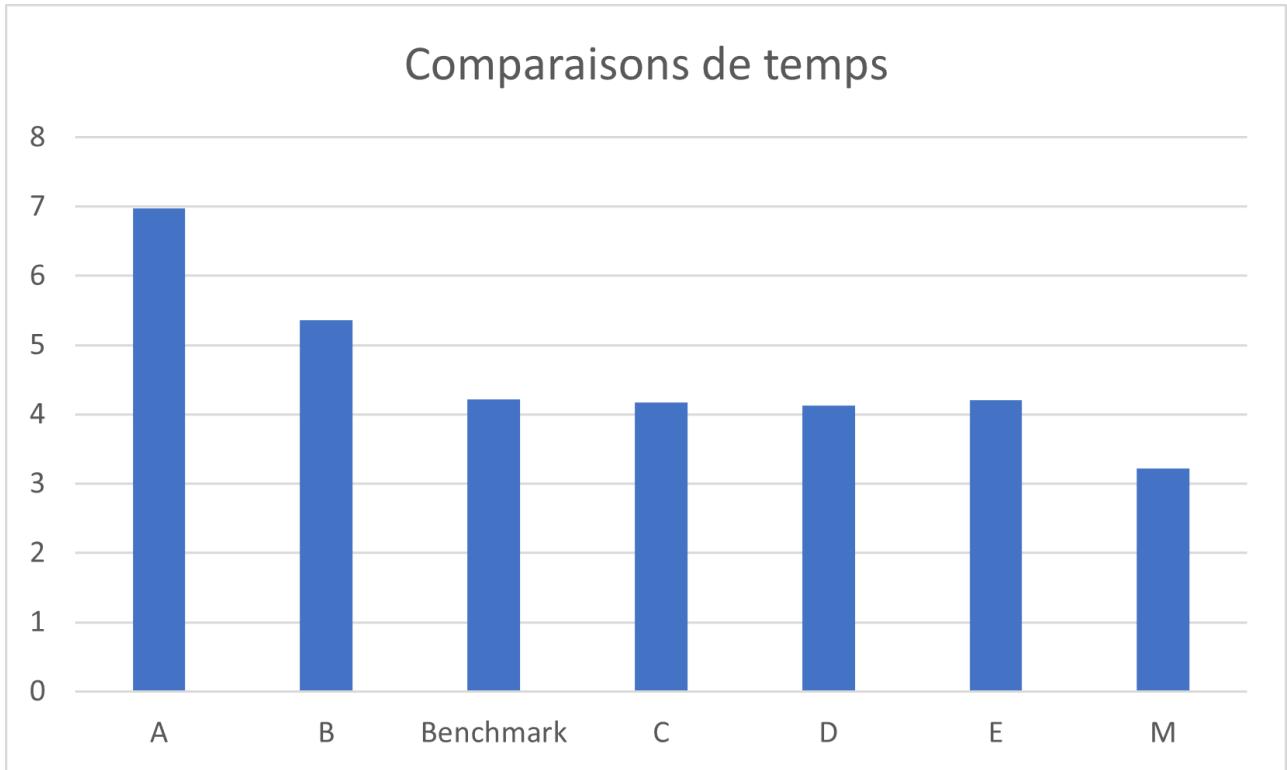
 $\downarrow$  $=$  $)$ 

## 2.5 Résultats

Comparer le programme benchmark avec d'autres versions du programme :

- Version A : Sans utiliser les heuristiques.
- Version B : Initialisation par Unit Matrix
- Version C : Ajouter la coupe de faisabilité
- Version D : Ajouter toutes les coupes
- Version E : Enhanced algorithm
- Version M : Intégrer la rounding heuristic

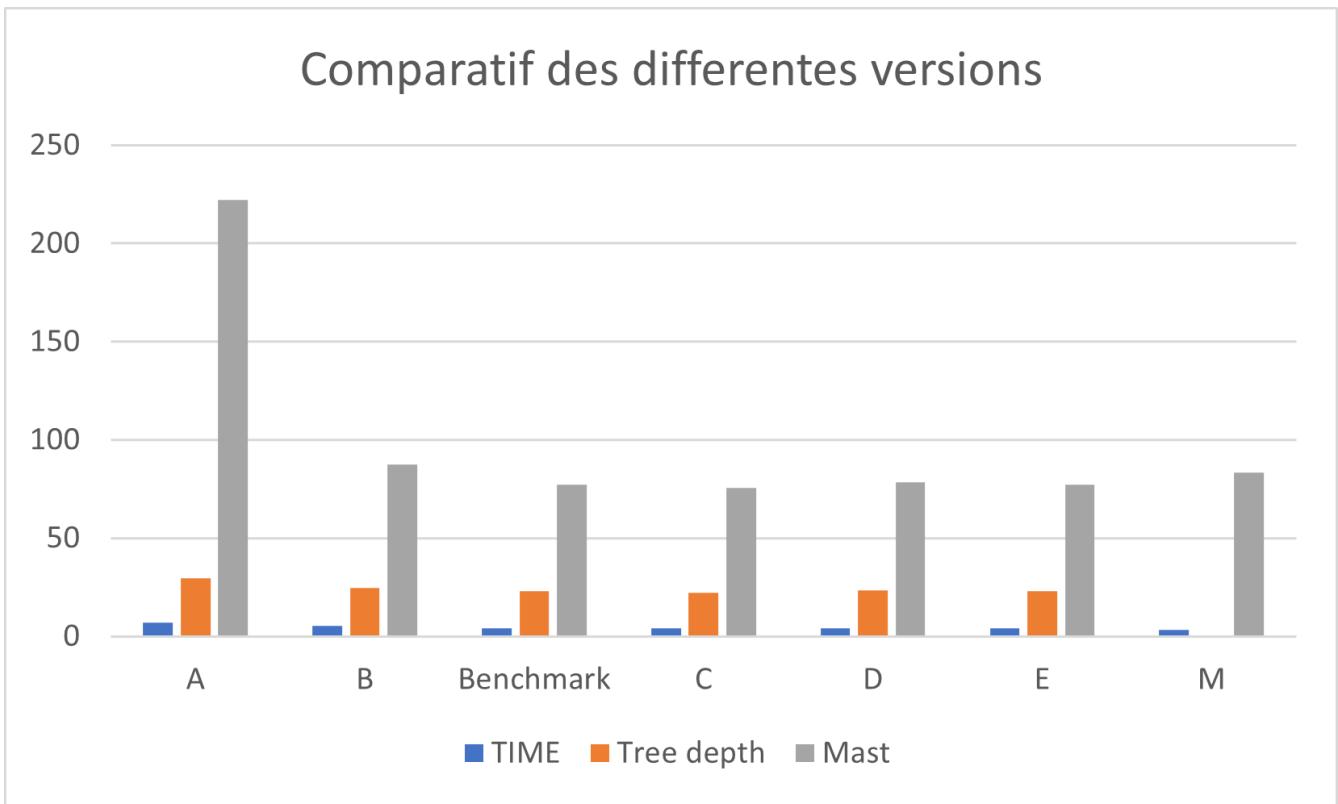
## 2.6 Résultats



La version M est la plus performante en moyenne



## 2.7 Résultats



# 07

# CONCLUSION



$$e = f^2(x+4gh)^2(s) \cdot (x)^3 \div (gh)^2 - x^2 \rightarrow$$

$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(h)e^3 + x^2 - 2x^2$$

$$g = x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h)$$

$$h = e^2 f g^2 - (x)^2 + (3)^2 (f)^3 + x(4x)$$

$$a = x(s^1) + (h)(c) + (d)(ef)^2 = x^2$$

$$(h)(d) \div (s^1)(h^2)(b)^2 = 4x^2 hd$$

$$x^3 \div (x)(x)^2 2x = 2s + 4x$$

$$x^2(h)$$

$$ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)^3}$$

$$x^2 + ab(s)^3$$

$$- (x)(s)^1$$



$$(x)^2 = ab$$

$$(x) = bc$$

# Merci De Votre Attention

