Optimisation sous Contraintes

Cyril Terrioux
cyril.terrioux@univ-amu.fr

Faculté des Sciences
Aix*Marseille Université

Plan

Plan

Contexte

Les instances CSP peuvent avoir beaucoup de solutions.

Contexte

Les instances CSP peuvent avoir beaucoup de solutions.

Toutes ne sont pas pertinentes.

Contexte

Les instances CSP peuvent avoir beaucoup de solutions.

Toutes ne sont pas pertinentes.

Nécessité d'utiliser un critère pour choisir les solutions pertinentes

Contexte

Les instances CSP peuvent avoir beaucoup de solutions.

Toutes ne sont pas pertinentes.

Nécessité d'utiliser un critère pour choisir les solutions pertinentes

Les instances issues du monde réel sont généralement des problèmes d'optimisation.

Applications

- Enchères combinatoires
- Bio-informatique
- Logistique
- Jeux
- Routage
- Transport
- Planification
- Télécommunication : Problème d'allocation de fréquences
- Théorie des graphes
- Agriculture
- Chimie
- ...

Plan

Problème d'optimisation sous contraintes (COP)

Instance $\mathcal{P} = (X, D, C, f)$ avec :

- $X = \{x_1, \ldots, x_n\}$ un ensemble de n variables,
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ un ensemble de domaines finis de taille au plus d,
- C un ensemble de m contraintes $c = (S(c), R(c))$:
  - $S(c) \subseteq X$: portée
  - $R(c) \subseteq \prod_{x \in S(c)} d_x$: relation
- f une fonction objectif impliquant les variables $x_{i_1}, \ldots, x_{i_m}$

Les critères d'optimisation

- Optimisation mono-critère :
  - max x
  - min $x + 2.y - z$
  - max $x.y.z$

Les critères d'optimisation

- Optimisation mono-critère :
  - max x
  - min $x + 2.y - z$
  - max $x.y.z$
- Optimisation multi-critères :

  Comment combiner les fonctions objectifs ?

Les critères d'optimisation

- Optimisation mono-critère :
  - max x
  - min $x + 2.y - z$
  - max $x.y.z$

- Optimisation multi-critères :

  Comment combiner les fonctions objectifs ?
  - ordre lexicographique
  - pareto

Solution optimale

Solution = affectation complète cohérente optimisant la fonction objectif

Trouver une affectation complète cohérente n'est donc plus suffisant !

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Modélisation :

- $X = \{x_1, \ldots, x_n\}$

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Modélisation :

- $X = \{x_1, \ldots, x_n\}$
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i,\ d_{x_i} = \{0, 1\}$

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Modélisation :

- $X = \{x_1, \ldots, x_n\}$
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i,\ d_{x_i} = \{0, 1\}$
- $C = \{c\}$ avec $S(c) = X$ et
  $R(c) = \{(w_1, \ldots, w_n) \in d_{x_1} \times \ldots \times d_{x_n}, \sum_i p_i.w_i \leq P\}$

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Modélisation :

- $X = \{x_1, \ldots, x_n\}$
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i,\ d_{x_i} = \{0, 1\}$
- $C = \{c\}$ avec $S(c) = X$ et
  $R(c) = \{(w_1, \ldots, w_n) \in d_{x_1} \times \ldots \times d_{x_n}, \sum_i p_i.w_i \leq P\}$
- Fonction objectif : $\max \sum_i v_i.x_i$

La contrainte Sum

Soit $Y = \{x_{i_1}, \ldots, x_{i_r}\}$

Soit $C = \{c_{i_1}, \ldots, c_{i_r}\}$

Soit un opérateur $\odot$ parmi $\{=, \neq, <, \leq, >, \geq, \in\}$

Soit k une valeur ou une variable

La contrainte Sum

Soit $Y = \{x_{i_1}, \dots, x_{i_r}\}$

Soit $C = \{c_{i_1}, \dots, c_{i_r}\}$

Soit un opérateur $\odot$ parmi $\{=, \neq, <, \leq, >, \geq, \in\}$

Soit k une valeur ou une variable

$Sum(Y, C, \odot, k)$ impose $\sum_{i_j} c_{i_j}.x_{i_j} \odot k$

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Exemple : Problème du sac à dos

Problème :

- Instance :
  - n objets chacun ayant un poids p_i et une valeur v_i
  - un sac à dos dont la charge maximale est P
- Objectif : maximiser la valeur des objets emportés tout en respectant la charge maximale

Modélisation :

- $X = \{x_1, \ldots, x_n\}$
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i,\ d_{x_i} = \{0, 1\}$
- $C = \{Sum(X, \{p_1, \ldots, p_n\}, \leq, P)\}$
- Fonction objectif : $\max \sum_i v_i . x_i$

Plan

Résolution

Taille de l'espace de recherche : $O(d^n)$

Résolution

Taille de l'espace de recherche : $O(d^n)$

- Méthodes complètes :
  - Développer des méthodes ad-hoc
  - Résoudre une succession d'instances CSP
- Méthodes incomplètes

Branch and Bound (BnB)

Utilisation de deux bornes :

- lb : minorant du coût de l'affectation courante
- ub : majorant de la valeur optimale de f (coût de la meilleure solution connue)

Branch and Bound (BnB)

Utilisation de deux bornes :

- lb : minorant du coût de l'affectation courante
- ub : majorant de la valeur optimale de f (coût de la meilleure solution connue)

Principe :

- on étend progressivement une affectation cohérente

Branch and Bound (BnB)

Utilisation de deux bornes :

- lb : minorant du coût de l'affectation courante
- ub : majorant de la valeur optimale de f (coût de la meilleure solution connue)

Principe :

- on étend progressivement une affectation cohérente
- en cas d'échec, on change la valeur de la variable courante

Branch and Bound (BnB)

Utilisation de deux bornes :

- *lb* : minorant du coût de l'affectation courante
- *ub* : majorant de la valeur optimale de f (coût de la meilleure solution connue)

Principe :

- on étend progressivement une affectation cohérente
- en cas d'échec, on change la valeur de la variable courante
- s'il n'y a plus de valeur, on revient sur la variable précédente

Branch and Bound (BnB)

Utilisation de deux bornes :

- lb : minorant du coût de l'affectation courante
- ub : majorant de la valeur optimale de f (coût de la meilleure solution connue)

Principe :

- on étend progressivement une affectation cohérente
- en cas d'échec, on change la valeur de la variable courante
- s'il n'y a plus de valeur, on revient sur la variable précédente
- on réitère le procédé tant qu'on n'a pas essayé toutes les possibilités et $lb < ub$

Branch and Bound (BnB)

BnB($\mathcal{A}, V$)

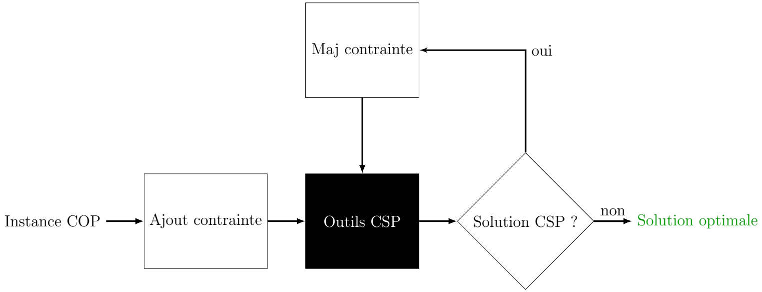
Si $V = \emptyset$
Alors $\mathcal{A}$ est une solution
    Mettre à jour *ub*
Sinon
  Choisir $x \in V$
  $d \leftarrow d_x$
  TantQue $d \neq \emptyset$
    Choisir v dans d
    $d \leftarrow d \backslash \{v\}$
    Si $\mathcal{A} \cup \{x = v\}$ est cohérente et $lb \leq ub$
    Alors BnB($\mathcal{A} \cup \{x = v\}, V \backslash \{x\}$)
    FinSi
  FinTantQue
FinSi

Branch and Bound (BnB) - Mise en œuvre

Exploitation des outils CSP

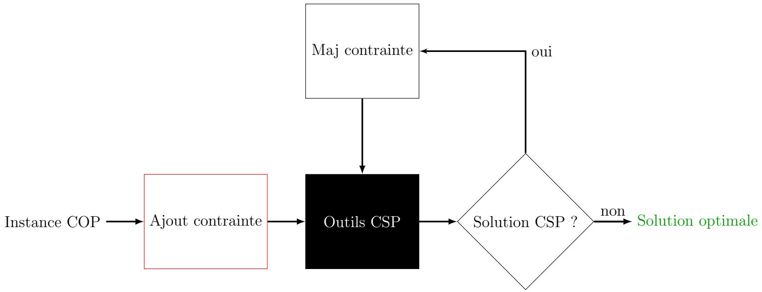
Branch and Bound (BnB) - Mise en œuvre

Exploitation des outils CSP



Branch and Bound (BnB) - Mise en œuvre

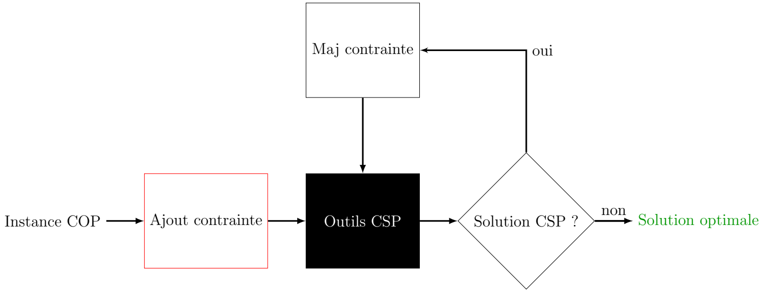
Exploitation des outils CSP



Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$

Branch and Bound (BnB) - Mise en œuvre

Exploitation des outils CSP

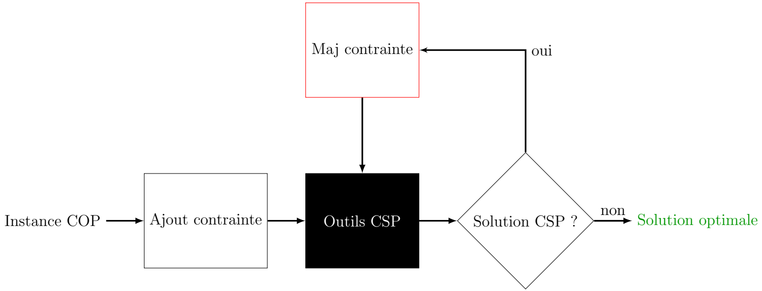


Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$
Ajout d'une contrainte $x_{opt} < ub_0$ avec

- $ub_0 = +\infty$ ou
- valeur calculée grâce à une méthode incomplète

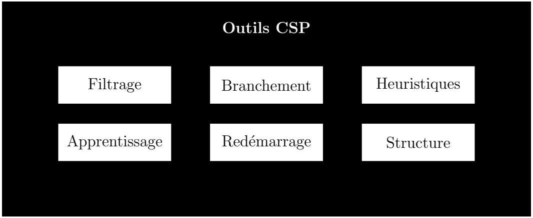
Branch and Bound (BnB) - Mise en œuvre

Exploitation des outils CSP

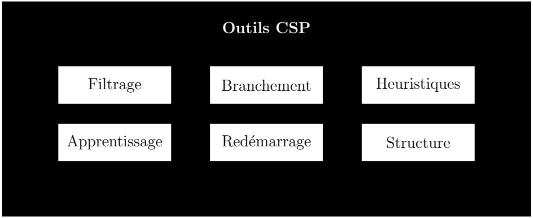


Mise à jour de la contrainte $x_{opt} < ub$ avec ub coût de la dernière solution

Branch and Bound (BnB) - Outils utilisés

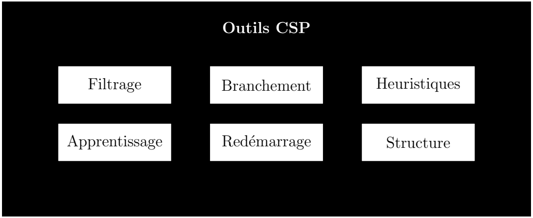


Branch and Bound (BnB) - Outils utilisés



Heuristiques de choix de valeurs plus importantes

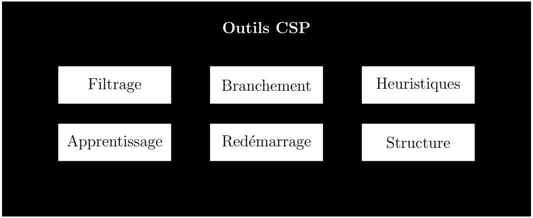
Branch and Bound (BnB) - Outils utilisés



Heuristiques de choix de valeurs plus importantes

Possibilité de trouver une solution CSP de coût intéressant plus vite

Branch and Bound (BnB) - Outils utilisés



Heuristiques de choix de valeurs plus importantes

Possibilité de trouver une solution CSP de coût intéressant plus vite

Réduction potentielle de l'espace de recherche visité

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

...

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

$\ldots$

kième solution CSP $\Rightarrow ub = 20$

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

...

kième solution CSP $\Rightarrow ub = 20$

Pas de solution de meilleur coût

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

...

kième solution CSP $\Rightarrow ub = 20$

Pas de solution de meilleur coût (preuve d'optimalité)

Branch and Bound (BnB) - Exemple

Initialement $ub = 150$

Première solution CSP $\Rightarrow ub = 100$

Deuxième solution CSP $\Rightarrow ub = 83$

Troisième solution CSP $\Rightarrow ub = 70$

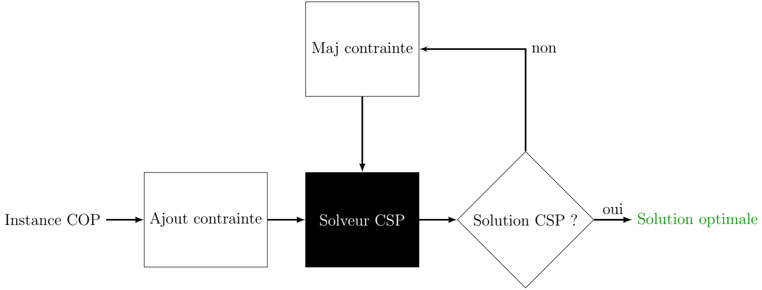
...

kième solution CSP $\Rightarrow ub = 20$

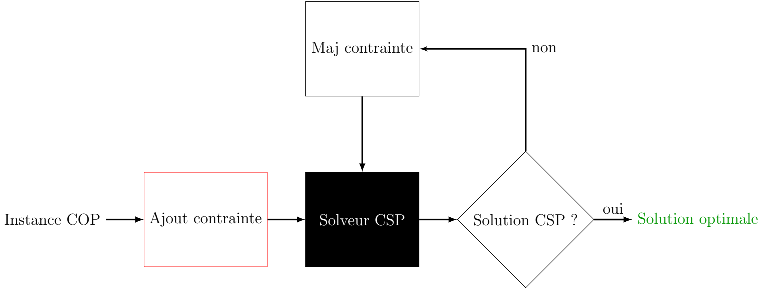
Pas de solution de meilleur coût (preuve d'optimalité)

Coût de la solution optimale : 20

Résolution itérative

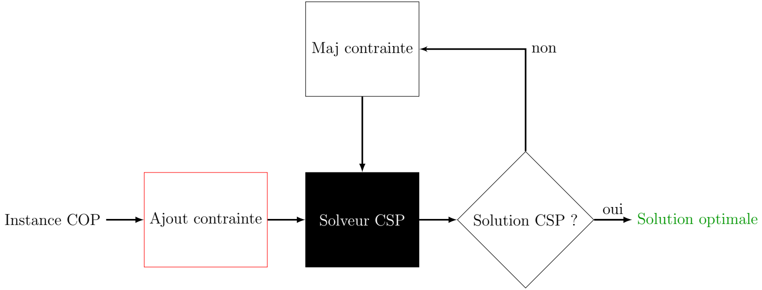


Résolution itérative



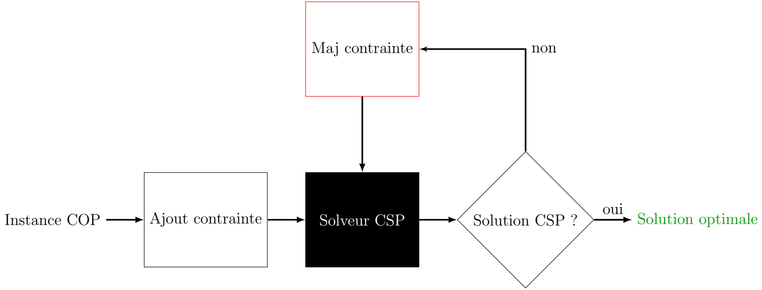
Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$

Résolution itérative



Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$
Ajout d'une contrainte $x_{opt} = k$ avec k un minorant calculé grâce à une méthode incomplète

Résolution itérative



Mise à jour de la contrainte $x_{opt} = k$ avec $k = k + 1$ s'il n'existe pas de solution

Résolution itérative - Exemple

Initialement $lb = 10$

Résolution itérative - Exemple

Initialement $lb = 10$

Existence d'une solution pour $x_{opt} = 10$: non

Résolution itérative - Exemple

Initialement $lb = 10$

Existence d'une solution pour $x_{opt} = 10$: non

Existence d'une solution pour $x_{opt} = 11$: non

Résolution itérative - Exemple

Initialement $lb = 10$

Existence d'une solution pour $x_{opt} = 10$: non

Existence d'une solution pour $x_{opt} = 11$: non

Existence d'une solution pour $x_{opt} = 12$: non

Résolution itérative - Exemple

Initialement $lb = 10$

Existence d'une solution pour $x_{opt} = 10$: non

Existence d'une solution pour $x_{opt} = 11$: non

Existence d'une solution pour $x_{opt} = 12$: non

...

Résolution itérative - Exemple

Initialement $lb = 10$

Existence d'une solution pour $x_{opt} = 10$: non

Existence d'une solution pour $x_{opt} = 11$: non

Existence d'une solution pour $x_{opt} = 12$: non

...

Existence d'une solution pour $x_{opt} = 20$: oui

Coût de la solution optimale : 20

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants
plusieurs solutions CSP	une seule solution CSP

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants
plusieurs solutions CSP	une seule solution CSP
une seule exploration	multitude de résolutions

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants
plusieurs solutions CSP	une seule solution CSP
une seule exploration	multitude de résolutions
$x_{opt} < k$	$x_{opt} = k$

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants
plusieurs solutions CSP	une seule solution CSP
une seule exploration	multitude de résolutions
$x_{opt} < k$	$x_{opt} = k$

Quelle approche est la meilleure ?

Branch and Bound vs résolution itérative

Utilisation des mêmes outils CSP

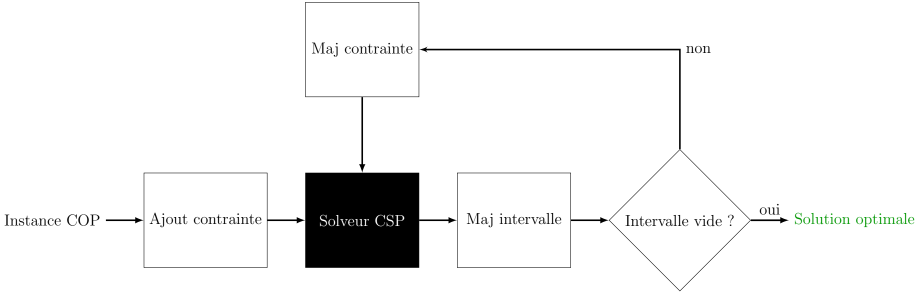
Mais des différences notables :

Branch and Bound	résolution itérative
décroissance de *ub*	croissance de *lb*
mise en œuvre d'un algorithme ad-hoc	exploitation de solveurs existants
plusieurs solutions CSP	une seule solution CSP
une seule exploration	multitude de résolutions
$x_{opt} < k$	$x_{opt} = k$

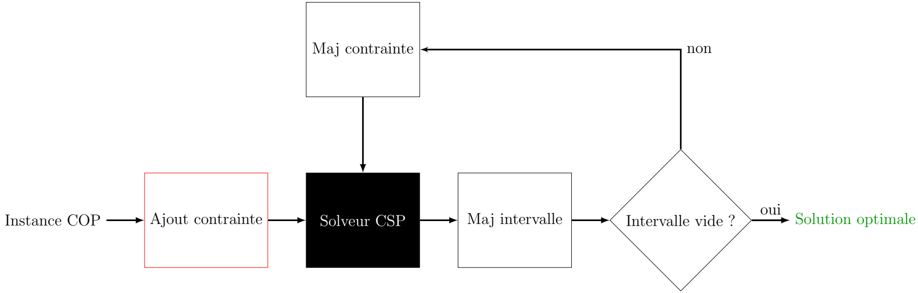
Quelle approche est la meilleure ?

Difficile à dire car dépendant de l'instance traitée

Approche par dichotomie

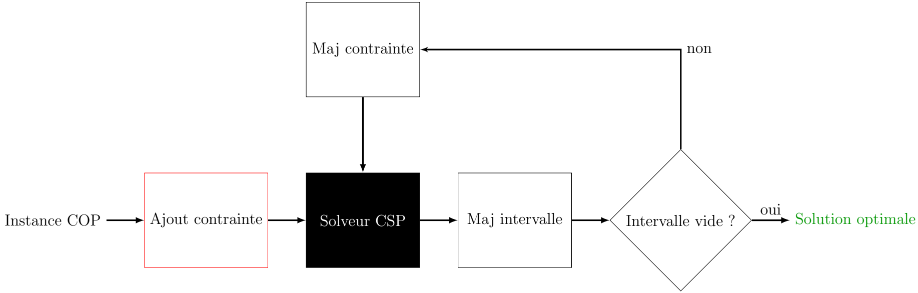


Approche par dichotomie



Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$

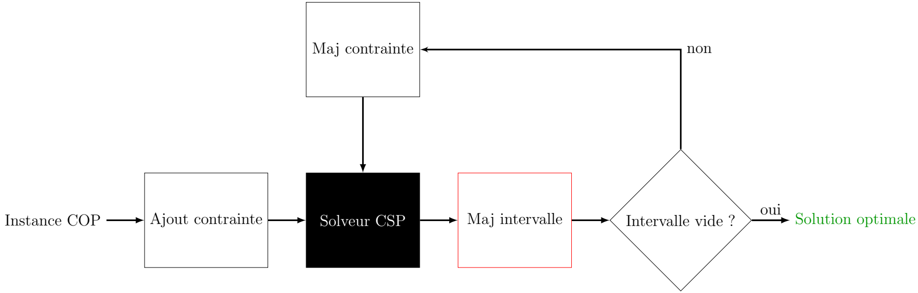
Approche par dichotomie



Ajout d'une variable x_{opt}
Ajout d'une contrainte $x_{opt} = f(x_{i_1}, \ldots, x_{i_m})$

$I = [lb_0, ub_0]$ avec lb_0 et ub_0 calculés grâce à une méthode incomplète
Ajout d'une contrainte $x_{opt} \in [lb_0, \frac{ub_0 + lb_0}{2}]$

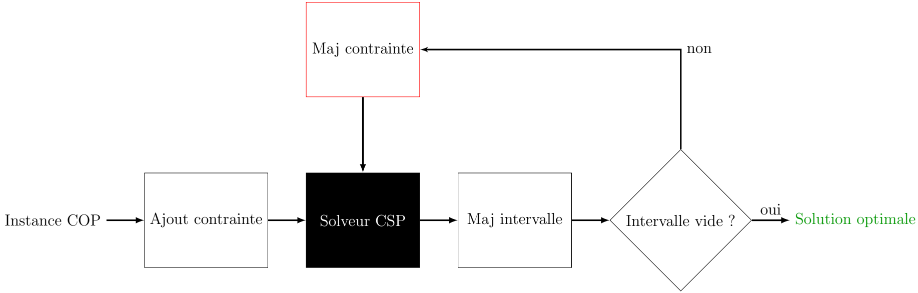
Approche par dichotomie



Mise à jour de l'intervalle $I = [lb, ub]$:

- $I = [lb, v - 1]$ si une solution de coût v a été trouvée
- $I = [\frac{ub+lb}{2} + 1, ub]$ sinon

Approche par dichotomie



Mise à jour de la contrainte $x_{opt} \in [lb, \frac{ub+lb}{2}]$

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

$I = [10, 150]$
Existence d'une solution pour $x_{opt} \in [10, 80]$: oui (solution de coût 71)

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

$I = [10, 150]$
Existence d'une solution pour $x_{opt} \in [10, 80]$: oui (solution de coût 71)

$I = [10, 70]$
Existence d'une solution pour $x_{opt} \in [10, 40]$: oui (solution de coût 40)

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

$I = [10, 150]$
Existence d'une solution pour $x_{opt} \in [10, 80]$: oui (solution de coût 71)

$I = [10, 70]$
Existence d'une solution pour $x_{opt} \in [10, 40]$: oui (solution de coût 40)

$I = [10, 39]$
Existence d'une solution pour $x_{opt} \in [10, 24]$: oui (solution de coût 23)

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

$I = [10, 150]$
Existence d'une solution pour $x_{opt} \in [10, 80]$: oui (solution de coût 71)

$I = [10, 70]$
Existence d'une solution pour $x_{opt} \in [10, 40]$: oui (solution de coût 40)

$I = [10, 39]$
Existence d'une solution pour $x_{opt} \in [10, 24]$: oui (solution de coût 23)

$I = [10, 22]$
Existence d'une solution pour $x_{opt} \in [10, 16]$: non

Approche par dichotomie - Exemple

Initialement $lb = 10$ et $ub = 150$

$I = [10, 150]$
Existence d'une solution pour $x_{opt} \in [10, 80]$: oui (solution de coût 71)

$I = [10, 70]$
Existence d'une solution pour $x_{opt} \in [10, 40]$: oui (solution de coût 40)

$I = [10, 39]$
Existence d'une solution pour $x_{opt} \in [10, 24]$: oui (solution de coût 23)

$I = [10, 22]$
Existence d'une solution pour $x_{opt} \in [10, 16]$: non

$I = [17, 22]$
Existence d'une solution pour $x_{opt} \in [17, 19]$: non

Approche par dichotomie - Exemple

$I = [20, 22]$

Existence d'une solution pour $x_{opt} \in [20, 21]$: oui (solution de coût 21)

Approche par dichotomie - Exemple

$I = [20, 22]$
Existence d'une solution pour $x_{opt} \in [20, 21]$: oui (solution de coût 21)

$I = [20, 20]$
Existence d'une solution pour $x_{opt} \in [20, 20]$: oui (solution de coût 20)

Approche par dichotomie - Exemple

$I = [20, 22]$
Existence d'une solution pour $x_{opt} \in [20, 21]$: oui (solution de coût 21)

$I = [20, 20]$
Existence d'une solution pour $x_{opt} \in [20, 20]$: oui (solution de coût 20)

$I = [20, 19] = \emptyset$

Coût de la solution optimale : 20

Quelques solveurs boîtes noires

De nombreux solveurs :

- AbsCon,
- Concrete,
- Choco,
- cosoco,
- Gecode,
- Mistral,
- OR-Tools,
- OscaR,
- PicatSAT,
- SAT4J,
- ...

Plan

Les enchères combinatoires

Problème :

- Instance :
  - un ensemble $O = \{o_1, \ldots, o_k\}$ d'objets,
  - un ensemble $E = \{e_1, \ldots, e_n\}$ d'enchères avec $e_i = (O_i, m_i)$ où :
    - $O_i \subseteq O$ l'ensemble d'objets souhaités par l'enchérisseur i
    - m_i le montant de l'enchère
- Objectif : trouver un sous-ensemble $E' \subseteq E$ t.q. aucune paire d'enchères ne partage d'objets et qui maximise la somme des montants

Les enchères combinatoires

Problème :

- Instance :
  - un ensemble $O = \{o_1, \ldots, o_k\}$ d'objets,
  - un ensemble $E = \{e_1, \ldots, e_n\}$ d'enchères avec $e_i = (O_i, m_i)$ où :
    - $O_i \subseteq O$ l'ensemble d'objets souhaités par l'enchérisseur i
    - m_i le montant de l'enchère
- Objectif : trouver un sous-ensemble $E' \subseteq E$ t.q. aucune paire d'enchères ne partage d'objets et qui maximise la somme des montants

Un problème très important du point de vue économique

Exemples :

- attribution des créneaux horaires des aéroports,
- vente de fréquences pour la téléphonie mobile,
- ...

Les enchères combinatoires

Modélisation :

- $X = \{x_1, \ldots, x_n\}$,
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i, d_{x_i} = \{0, 1\}$

Les enchères combinatoires

Modélisation :

- $X = \{x_1, \ldots, x_n\}$,
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i, d_{x_i} = \{0, 1\}$
- $C = \{c_{ij} | O_i \cap O_j \neq \emptyset\}$ avec :
  - $S(c_{ij}) = \{x_i, x_j\}$
  - $R(c_{ij}) = \{(v_i, v_j) \in d_{x_i} \times d_{x_j} | v_i.v_j = 0\}$

Les enchères combinatoires

Modélisation :

- $X = \{x_1, \ldots, x_n\}$,
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i, d_{x_i} = \{0, 1\}$
- $C = \{c_{ij} | O_i \cap O_j \neq \emptyset\}$ avec :
  - $S(c_{ij}) = \{x_i, x_j\}$
  - $R(c_{ij}) = \{(v_i, v_j) \in d_{x_i} \times d_{x_j} | v_i.v_j = 0\} = \{(0,0), (0,1), (1,0)\}$

Les enchères combinatoires

Modélisation :

- $X = \{x_1, \ldots, x_n\}$,
- $D = \{d_{x_1}, \ldots, d_{x_n}\}$ avec $\forall i, d_{x_i} = \{0, 1\}$
- $C = \{c_{ij} | O_i \cap O_j \neq \emptyset\}$ avec :
  - $S(c_{ij}) = \{x_i, x_j\}$
  - $R(c_{ij}) = \{(v_i, v_j) \in d_{x_i} \times d_{x_j} | v_i.v_j = 0\} = \{(0,0), (0,1), (1,0)\}$
- Fonction objectif : $\max \sum_i m_i.x_i$

Localisation d'entrepôts

Problème :

- Instance :
  - un ensemble $E = \{e_1, \ldots, e_n\}$ d'entrepôts avec :
    - n_i : nombre maximum de magasins livrables par l'entrepôt i
    - cm_i : coût de maintenance
  - un ensemble $M = \{m_1, \ldots, m_k\}$ de magasins :
    - ca_{ij} : coût d'approvisionnement du magasin i par l'entrepôt j
- Objectif : trouver un sous-ensemble $E' \subseteq E$ t.q. chaque magasin soit approvisionné par exactement un entrepôt tout en minimisant les coûts

La contrainte Element

Soit $Y = \{x_{i_1}, \ldots, x_{i_r}\}$

Soit v une valeur ou une variable

$Element(Y, v)$ impose qu'il existe un entier j t.q. $x_{i_j} = v$

La contrainte Element

Soit $Y = \{x_{i_1}, \ldots, x_{i_r}\}$

Soit v une valeur ou une variable

$Element(Y, v)$ impose qu'il existe un entier j t.q. $x_{i_j} = v$

Soit j une valeur ou une variable

$Element(Y, j, v)$ impose que $x_{i_j} = v$

La contrainte Count

Soit $Y = \{x_{i_1}, \ldots, x_{i_r}\}$

Soit un ensemble de valeurs V

Soit un opérateur $\odot$ parmi $\{=, \neq, <, \leq, >, \geq\}$

Soit un entier k

$Count(Y, V, \odot, k)$ impose que $|\{1 \leq j \leq r | x_{i_j} \in V\}| \odot k$

Localisation d'entrepôts

Modélisation :

- $X = \{m_1, \dots, m_k\} \cup \{o_1, \dots, o_n\} \cup \{c_1, \dots, c_k\}$,
- $D = \{d_{m_1}, \dots, d_{m_k}\} \cup \{d_{o_1}, \dots, d_{o_n}\} \cup \{d_{c_1}, \dots, d_{c_k}\}$ avec :
  - $\forall i, d_{m_i} = \{1, \dots, n\}$
  - $\forall i, d_{o_i} = \{0, 1\}$
  - $\forall i, d_{c_i} = \{ca_{i1}, \dots, ca_{in}\}$

Localisation d'entrepôts

Modélisation :

- $X = \{m_1, \ldots, m_k\} \cup \{o_1, \ldots, o_n\} \cup \{c_1, \ldots, c_k\}$,
- $D = \{d_{m_1}, \ldots, d_{m_k}\} \cup \{d_{o_1}, \ldots, d_{o_n}\} \cup \{d_{c_1}, \ldots, d_{c_k}\}$ avec :
  - $\forall i, d_{m_i} = \{1, \ldots, n\}$
  - $\forall i, d_{o_i} = \{0, 1\}$
  - $\forall i, d_{c_i} = \{ca_{i1}, \ldots, ca_{in}\}$
- $C = \{\ldots\}$ avec :
  - $\forall i \in \{1, \ldots, n\},\ Count(\{m_1, \ldots, m_k\}, i, \leq, n_i)$
  - $\forall i \in \{1, \ldots, k\},\ Element(\{o_1, \ldots, o_n\}, m_i, 1)$
  - $\forall i \in \{1, \ldots, k\},\ Element(\{ca_{i1}, \ldots, ca_{in}\}, m_i, c_i)$

Localisation d'entrepôts

Modélisation :

- $X = \{m_1, \ldots, m_k\} \cup \{o_1, \ldots, o_n\} \cup \{c_1, \ldots, c_k\}$,
- $D = \{d_{m_1}, \ldots, d_{m_k}\} \cup \{d_{o_1}, \ldots, d_{o_n}\} \cup \{d_{c_1}, \ldots, d_{c_k}\}$ avec :
  - $\forall i, d_{m_i} = \{1, \ldots, n\}$
  - $\forall i, d_{o_i} = \{0, 1\}$
  - $\forall i, d_{c_i} = \{ca_{i1}, \ldots, ca_{in}\}$
- $C = \{\ldots\}$ avec :
  - $\forall i \in \{1, \ldots, n\},\ Count(\{m_1, \ldots, m_k\}, i, \leq, n_i)$
  - $\forall i \in \{1, \ldots, k\},\ Element(\{o_1, \ldots, o_n\}, m_i, 1)$
  - $\forall i \in \{1, \ldots, k\},\ Element(\{ca_{i1}, \ldots, ca_{in}\}, m_i, c_i)$
- Fonction objectif : $\min \sum_{i=1}^{k} c_i + \sum_{i=1}^{n} cm_i.o_i$

Maintenance de centrales électriques

Problème :

- Instance :
  - un ensemble $P = \{1, \ldots, p\}$ de périodes
  - un ensemble de n unités de production avec :
    - p_i : puissance fournie par l'unité i
    - m_{ij} : coût de la maintenance de l'unité i pendant la période j
    - u_{ij} : coût de l'utilisation pendant la période j
    - a_i : période à laquelle la maintenance peut commencer au plus tôt
    - b_i : période à laquelle la maintenance doit commencer au plus tard
    - d_i : durée de la maintenance
  - D_j la demande électrique prévue pour la période j
- Objectif : trouver un étalement des maintenances garantissant un approvisionnement électrique suffisant tout en minimisant les coûts

Maintenance de centrales électriques

Modélisation :

- $X = \{x_{11}, \ldots, x_{np}\} \cup \{y_1, \ldots, y_n\} \cup \{z_1, \ldots, z_n\}$,

Maintenance de centrales électriques

Modélisation :

- $X = \{x_{11}, \ldots, x_{np}\} \cup \{y_1, \ldots, y_n\} \cup \{z_1, \ldots, z_n\}$,
- $D = \{d_{x_{11}}, \ldots, d_{x_{np}}\} \cup \{d_{y_1}, \ldots, d_{y_n}\} \cup \{d_{z_1}, \ldots, d_{z_n}\}$ avec :
  - $\forall i, j, \; d_{x_{ij}} = \{0, 1\}$
  - $\forall i, \; d_{y_i} = \{a_i, \ldots, b_i\}$
  - $\forall i, \; d_{z_i} = \{1, \ldots, p\}$

Maintenance de centrales électriques

Modélisation :

- $X = \{x_{11}, \ldots, x_{np}\} \cup \{y_1, \ldots, y_n\} \cup \{z_1, \ldots, z_n\}$,
- $D = \{d_{x_{11}}, \ldots, d_{x_{np}}\} \cup \{d_{y_1}, \ldots, d_{y_n}\} \cup \{d_{z_1}, \ldots, d_{z_n}\}$ avec :
  - $\forall i, j,\ d_{x_{ij}} = \{0, 1\}$
  - $\forall i,\ d_{y_i} = \{a_i, \ldots, b_i\}$
  - $\forall i,\ d_{z_i} = \{1, \ldots, p\}$
- $C = \{\ldots\}$:
  - Production suffisante : $\forall j,\ \sum_i p_i.x_{ij} \geq D_j$
  - Maintenance :
    - $\forall i, j,\ y_i = j \rightarrow x_{ij} = 0$
    - $\forall i, j,\ z_i = j \rightarrow x_{ij} = 1$
    - $\forall i, j,\ x_{ij} = 0 \wedge z_i > j + 1 \rightarrow x_{ij+1} = 0$
    - $\forall i, z_i = y_i + d_i$
    - $\forall i, j,\ x_{ij} = 0 \rightarrow y_i \leq j < z_i$
    - $\forall i, j,\ x_{ij} = 1 \rightarrow j < y_i \vee z_i \leq j$

Maintenance de centrales électriques

Modélisation :

- $X = \{x_{11}, \ldots, x_{np}\} \cup \{y_1, \ldots, y_n\} \cup \{z_1, \ldots, z_n\}$,
- $D = \{d_{x_{11}}, \ldots, d_{x_{np}}\} \cup \{d_{y_1}, \ldots, d_{y_n}\} \cup \{d_{z_1}, \ldots, d_{z_n}\}$ avec :
  - $\forall i, j,\ d_{x_{ij}} = \{0, 1\}$
  - $\forall i,\ d_{y_i} = \{a_i, \ldots, b_i\}$
  - $\forall i,\ d_{z_i} = \{1, \ldots, p\}$
- $C = \{\ldots\}$:
  - Production suffisante : $\forall j,\ \sum_i p_i.x_{ij} \geq D_j$
  - Maintenance :
    - $\forall i, j,\ y_i = j \rightarrow x_{ij} = 0$
    - $\forall i, j,\ z_i = j \rightarrow x_{ij} = 1$
    - $\forall i, j,\ x_{ij} = 0 \wedge z_i > j + 1 \rightarrow x_{ij+1} = 0$
    - $\forall i, z_i = y_i + d_i$
    - $\forall i, j,\ x_{ij} = 0 \rightarrow y_i \leq j < z_i$
    - $\forall i, j,\ x_{ij} = 1 \rightarrow j < y_i \vee z_i \leq j$
- Fonction objectif : $\min \sum_j \sum_i (u_{ij}.x_{ij} + m_{ij}.(1 - x_{ij}))$

Maintenance de centrales électriques

Deuxième modélisation :

- $X = \{x_{11}, \ldots, x_{np}, \} \cup \{y_1, \ldots, y_n\}$,

Maintenance de centrales électriques

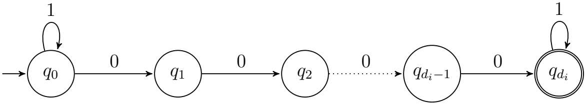
Deuxième modélisation :

- $X = \{x_{11}, \ldots, x_{np}, \} \cup \{y_1, \ldots, y_n\}$,
- $D = \{d_{x_{11}}, \ldots, d_{x_{np}}\} \cup \{d_{y_1}, \ldots, d_{y_n}\}$ avec :
  - $\forall i, j,\ d_{x_{ij}} = \{0, 1\}$
  - $\forall i,\ d_{y_i} = \{a_i, \ldots, b_i\}$

Maintenance de centrales électriques

Deuxième modélisation :

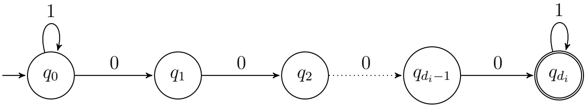
- $X = \{x_{11}, \ldots, x_{np},\} \cup \{y_1, \ldots, y_n\}$,
- $D = \{d_{x_{11}}, \ldots, d_{x_{np}}\} \cup \{d_{y_1}, \ldots, d_{y_n}\}$ avec :
  - $\forall i, j,\ d_{x_{ij}} = \{0, 1\}$
  - $\forall i,\ d_{y_i} = \{a_i, \ldots, b_i\}$
- $C = \{\ldots\}$:
  - Production suffisante : $\forall j,\ \sum_i p_i.x_{ij} \geq D_j$
  - Maintenance :
    - $\forall i, j,\ y_i = j \rightarrow x_{ij} = 0$
    - $\forall i, j,\ x_{ij} = 0 \rightarrow y_i \leq j$
    - $\forall i,\ Regular(\{x_{i1}, x_{i2}, \ldots, x_{ip}\}, A)$



Maintenance de centrales électriques

Deuxième modélisation :

- $X = \{x_{11}, \dots, x_{np},\} \cup \{y_1, \dots, y_n\}$,
- $D = \{d_{x_{11}}, \dots, d_{x_{np}}\} \cup \{d_{y_1}, \dots, d_{y_n}\}$ avec :
  - $\forall i, j,\ d_{x_{ij}} = \{0, 1\}$
  - $\forall i,\ d_{y_i} = \{a_i, \dots, b_i\}$
- $C = \{\dots\}$:
  - Production suffisante : $\forall j,\ \sum_i p_i.x_{ij} \geq D_j$
  - Maintenance :
    - $\forall i, j,\ y_i = j \to x_{ij} = 0$
    - $\forall i, j,\ x_{ij} = 0 \to y_i \leq j$
    - $\forall i,\ Regular(\{x_{i1}, x_{i2}, \dots, x_{ip}\}, A)$



- Fonction objectif : $\min \sum_j \sum_i (u_{ij}.x_{ij} + m_{ij}.(1 - x_{ij}))$