Méthodes et modélisation pour l'optimisation

M1 informatique, 2016–2017

06 — Modélisation SAT et DPLL

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

INSTITUT D'ÉLECTRONIQUE ET D'INFORMATIQUE GASPARD-MONGE

Emploi du temps

	Cours	Enseignant
i=1	MMPO	Thapper
2	Complexité	Nicaud
3	MMPO	Hubard
	Compression	Nicaud
	...	...

	Lun	Mar	Mer	Jeu	Ven
08h30 10h30	j=1				
10h45 12h45	2				
14h00 16h00	3				
16h15 18h15					

Dis	
Thapper	lun, mar, mer 10h45-16h00
Hubard	lun, ven
...	..

Séances i = 1, ..., n

Créneaux j = 1, ..., m

Emploi du temps

- Variables x_{ij}

  intention : x_{ij} = 1 si la séance i est affectée au créneau j

1. La séance $i = 1, \ldots, n$ doit avoir lieu au plus une fois :

   $$\text{AtMostOne}(x_{i1}, \ldots, x_{im})$$

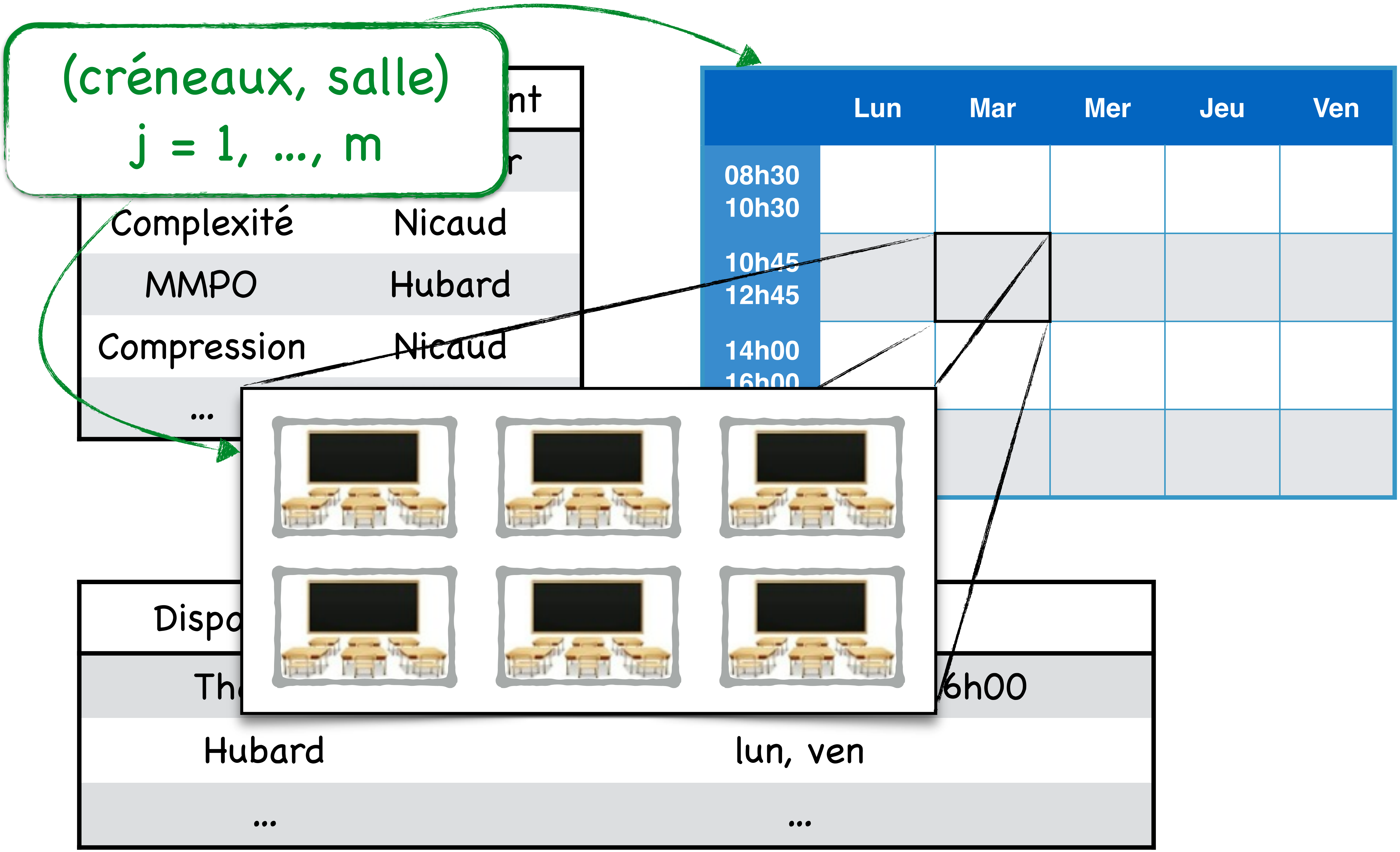
2. Le créneau $j = 1, \ldots, m$ doit être affecté à au plus une séance :

   $$\text{AtMostOne}(x_{1j}, \ldots, x_{nj})$$

3. Soit $j_1, \ldots, j_d$ les disponibilités de l'enseignant de la séance $i = 1, \ldots, n$.
   La séance i doit avoir lieu sur un de ces créneaux :

   $$\text{AtLeastOne}(x_{ij1}, \ldots, x_{ijd})$$

Emploi du temps 2



Emploi du temps 2

- Variables x_{ij}
  intention : x_{ij} = 1 si la séance i est affectée à **la paire** (créneau,salle) j

1. La séance $i = 1, \ldots, n$ doit avoir lieu au plus une fois :

   $$\text{AtMostOne}(x_{i1}, \ldots, x_{im})$$

2. La paire (créneau,salle) $j = 1, \ldots, m$ doit être affectée à au plus une séance :

   $$\text{AtMostOne}(x_{1j}, \ldots, x_{nj})$$

3. Soit $j_1, \ldots, j_d$ les disponibilités de l'enseignant de la séance $i = 1, \ldots, n$.
   La séance i doit avoir lieu sur un de ces créneaux :

   $$\text{AtLeastOne}(x_{ij_1}, \ldots, x_{ij_d})$$

Emploi du temps 2

- Variables x_{ij}

  intention : $x_{ij} = 1$ si la séance i est affectée à **la paire** (créneau,salle) j

4. Pour toute paire de séances i_1, i_2 qui sont assurées par le même enseignant et toute paire j_1, j_2 où j_1 et j_2 ont lieu à la même horaire (créneau), on a

$$\text{AtMostOne}(x_{i1j1},\ x_{i2j2}) \Leftrightarrow (\neg x_{i1j1} \vee \neg x_{i2j2})$$

Algorithme DPLL

- Entrée : une formule en CNF
- Sortie : **SAT** si la formule est satisfaisable, **UNSAT** sinon
- Algorithme naïf : essayer toutes les 2^n affectations

$$f : \{x1, ..., xn\} \rightarrow \{0,1\}$$

- Problème NP-complet — en général, on ne peut pas faire (beaucoup) mieux.
- Amélioration : étape de raisonnement ajoutée pour éviter de considérer toutes les possibilités.

Propagation unitaire

$$(x_P),\ (\neg x_N),\ (\neg x_J \lor x_N),\ (x_P \lor x_J)$$

Si on veut satisfaire toutes les clauses, il faut satisfaire la clause (unitaire) (x_P)

$$\cancel{(x_P)},\ (\neg x_N),\ (\neg x_J \lor x_N),\ \cancel{(x_P \lor x_J)}$$

$$(\neg x_N),\ (\neg x_J \lor x_N)$$

Si on veut satisfaire toutes les clauses, il faut satisfaire la clause (unitaire) $(\neg x_N)$

$$\cancel{(\neg x_N)},\ (\neg x_J \lor \cancel{x_N})$$

$$(\neg x_J)$$

Si on veut satisfaire toutes les clauses, il faut satisfaire la clause (unitaire) $(\neg x_J)$

$$\cancel{(\neg x_J)}$$

C'est fini ! Toutes les clauses sont satisfaites

Affectation partielle : $x_P = 1 \quad x_N = 0 \quad x_J = 0$

Propagation unitaire

- Une **clause unitaire** est une clause (u) qui ne contient qu'un seul littéral ; u = x ou u = ¬x
- ( ) note **la clause vide**
- Une clause et satisfaite ssi au moins un littéral est vrai, donc ( ) est toujours **fausse**
- Une formule est satisfaite ssi toutes ses clauses sont satisfaites, donc la formule vide, { }, est toujours **vraie**

PU

- ***Entrée*** : un ensemble F de clauses
- ***Sortie*** : un ensemble F' de clauses ;

  F' étant satisfaisable ssi F est satisfaisable

$F' \longleftarrow F$

While F' contient une clause unitaire (u)

  supprimer toutes ***occurrences*** de $\neg u$ de F'

  supprimer toutes ***clauses*** qui contiennent u de F'

Return F'

Exemple PU

Effectuer la propagation unitaire dans l'ensemble de clauses suivant

$$\cancel{(x1)},\ \cancel{(x1 \vee x2 \vee \neg x3)},\ \cancel{(\neg \cancel{x1} \vee x3 \vee x4)},\ \cancel{(\neg \cancel{x1} \vee \neg x2)},$$
$$\cancel{(\neg \cancel{x1} \vee \neg x2 \vee x3)},\ (x3 \vee \neg \cancel{x4} \vee x5),\ \cancel{(\neg \cancel{x1} \vee \cancel{x2} \vee x4)}$$

On supprime une occurrence de ¬u avec $(\cancel{x}\ \ldots)$

On supprime une clause qui contient u avec $\cancel{(\qquad)}$

Clause unitaire : (x1) (¬x2) (x4)

Affectation : x1 = 1 x2 = 0 x4 = 1

Clauses restantes : (x3 ∨ x5)

Exemple PU

Effectuer la propagation unitaire dans l'ensemble de clauses suivant

$\cancel{(x1)}, \cancel{(\neg x1 \vee x2)}, (\neg \cancel{x}1 \vee x3 \vee x4), (\neg \cancel{x}1 \vee \neg \cancel{x}2)$

On supprime une occurrence de $\neg u$ avec $(\, \cancel{x} \ldots)$

On supprime une clause qui contient u avec $\cancel{(\qquad)}$

Clause unitaire : $(x1)$ $(x2)$

Affectation : $x1 = 1$ $x2 = 1$

Clause vide, donc l'ensemble n'est pas satisfaisable !

Clauses restantes : $(x3 \vee x4), (\ )$

A retenir

- Il peut y avoir un choix entre deux clauses unitaires différentes. Dans ce cas, l'ordre de choix n'a pas d'importance : le résultat sera le même.
- La PU n'affecte pas toujours toutes les variables.
- Si la PU rend une clause vide, alors l'ensemble de clauses restantes n'est pas satisfaisable, donc l'ensemble de clauses d'origine n'était pas satisfaisable non plus.
- Donc, la PU peut :
  - trouver une affectation satisfaisante
  - trouver une affectation partielle
  - assurer que l'ensemble de clauses n'est pas satisfaisable

DPLL

- ***Entrée*** : un ensemble F de clauses
- ***Sortie*** : "SAT" ou "UNSAT"

F $\longleftarrow$ **PU**(F)

If F contient la clause vide, **then**

    Return "UNSAT"

If F est vide, **then**

    Return "SAT"

x $\longleftarrow$ une variable non-affectée

If **DPLL**(F ∪ {(x)}) == "SAT", **then**

    Return "SAT"

Else

    Return **DPLL**(F ∪ {(¬x)})

Exemple DPLL

Résoudre la formule suivante par l'algorithme DPLL

$F = \{(x1 \lor \neg x2 \lor \neg x3), (\neg x1 \lor \neg x2 \lor \neg x3), (\neg x1 \lor x2 \lor \neg x3), (x1 \lor x3)\}$

- **Pile : vide**

  $F_1 \longleftarrow PU(F)$ ne modifie rien

  Choix de variable : x1

- **Pile : x1 = 1**

  $F_2 \longleftarrow PU(F_1 \cup \{(x1)\}) = \{\cancel{(x1 \lor \neg x2 \lor \neg x3)}, (\cancel{\neg x1} \lor \neg x2 \lor \neg x3), (\cancel{\neg x1} \lor x2 \lor \neg x3), \cancel{(x1 \lor x3)}, \cancel{(x1)}\}$

  Choix de variable : x2

- **Pile : x1 = 1, x2 = 1**

  $F_3 \longleftarrow PU(F_2 \cup \{(x2)\}) = \{\cancel{(\cancel{\neg x2} \lor \neg x3)}, \cancel{(x2 \lor \neg x3)}, \cancel{(x2)}\}$

  PU affecte : x3 = 0

  $F_3 = \{\ \}$, donc la formule est satisfaite par : x1 = 1, x2 = 1, x3 = 0

Exemples

Résoudre les formules suivantes par l'algorithme DPLL

$F = \{ (\neg x1 \vee x2), (\neg x1 \vee \neg x2 \vee \neg x3), (\neg x2 \vee x3) \}$

$F = \{ (x1 \vee x2), (\neg x1 \vee \neg x2), (\neg x2 \vee x3), (\neg x2 \vee \neg x3),$
$(x3 \vee x4), (\neg x4 \vee x5), (\neg x3 \vee x4), (\neg x4 \vee \neg x5) \}$

A retenir

- Différentes règles de choix de variable possibles
  - Nous utilisons l'ordre lexicographique : x1, …, xn et testons d'abord l'affectation à 1, puis à 0
- Sur la pile, on met les affectations faites par la PU entre crochets :
  - **Pile : x1 = 1, [x2 = 1], x3 = 1**
- En trouvant une clause vide dans F, on "backtrack" :
  - on dépile jusqu'à une affectation x = 1 (faite par un choix de variable, non par la PU) et on la remplace par x = 0 ;
  - si on vide la pile, on renvoie "UNSAT"

Examen le 19 janvier

- Sur les notions abordées aux cours, TD et TP
- 1 feuille recto-verso manuscrite autorisée
- Téléphone, calculatrice, machine interdits