

---

# OPC - Modélisation en variables booléennes

---

Vous pouvez utiliser **Sat4j** pour résoudre l'ensemble de ces problèmes. Comme les formats d'entrée que nous utilisons sont standards, vous pouvez aussi utiliser d'autres prouveurs. Si vous utilisez une machine sous Linux, les prouveurs SAT **minisat**, **picosat** et **lingeling** doivent être disponibles pour votre distribution favorite. Le prouveur SAT/PB/MAXSAT **clasp** est normalement disponible sur toutes les distributions Linux.

## 1 Modélisation Pseudo-Booléenne

### 1.1 Subset sum problem

Soient  $P$  un ensemble d'entiers et un entier  $k$ . On cherche  $E \subseteq P$  tel que  $\sum_{e \in E} e = k$ .

- Modéliser ce problème sous la forme de problème pseudo-booléen.
- En utilisant **le format d'entrée de la compétition pseudo-booléenne**, utiliser un prouveur pseudo booléen pour résoudre le problème suivant :  $P = \{2, 35, 47, 68, 89, 120, 146, 179, 291, 303, 459, 659\}$  avec  $k = 567$  puis  $k = 693$ .

### 1.2 Knapsack

Reprendre l'exercice 3 du TP 2, knapsack, en utilisant cette fois ci un codage pseudo-booléen. On utilisera les mêmes données et on vérifiera la concordance des solutions obtenues à l'aide de la modélisation CSP et de la modélisation pseudo-booléenne.

*Attention : il faudra transformer ce problème de maximisation en un problème de minimisation pour le traduire au format pseudo-booléen.*

## 2 Modélisation en SAT/Pseudo/Maxsat

Nous allons maintenant nous intéresser au problème de gestion de dépendances logicielles. Le fichier CUDF suivant présente un exemple de déclaration de paquets comme on peut les trouver sous Linux. Chaque paquet est disponible en une ou deux versions. Ces versions peuvent être incompatibles. Les paquets ont des dépendances : pour installer **evolution** version 3, il faut installer **gnome** version 3. Les dépendances peuvent être déclarées implicitement à l'aide de l'attribut *provides* : pour installer **web-machine** (version 1 ou 2) il faudra installer **firefox 2** ou **firefox 3** ou **chrome 10** ou **opera 11**. L'utilisateur souhaite installer une **web-machine** (il ne précise pas la version).

- Modéliser en logique propositionnelle ce problème et **utiliser le format Dimacs de la compétition SAT** pour créer un fichier CNF utilisable par un prouveur SAT. Vérifier avec un prouveur SAT que vous obtenez bien une installation qui satisfait ces dépendances.
- On souhaite maintenant minimiser la taille de l'installation (somme des attributs *installsize*). Modéliser en pseudo-booléen ce problème. On notera que la clause  $a \vee b \vee c$  s'écrit  $a + b + c \geq 1$  en PB, que  $\neg a \vee b$  s'écrit  $-a + b \geq 0$ , et que  $\neg a \vee \neg b$  s'écrit  $-a - b \geq -1$ .
- On souhaite cette fois ci maximiser la taille de l'installation. Que doit-on observer ? Utiliser le formalisme Partial Weighted MaSat pour représenter ce nouveau problème d'optimisation.

- Quelles sont les clauses dures (qu'il faut nécessairement satisfaire)?
  - Quelles sont les clauses souples (que l'on peut relaxer)? Que va représenter leur poids?
- On utilisera le format de la compétition MaxSat pour écrire un fichier au format wcnf qui étend le format Dimacs au cadre MaxSat.
- On souhaite maintenant s'assurer que l'on installe en priorité la dernière version d'un logiciel. Comment pourrait-on procéder? Quel problème rencontre-t-on?

### Un problème de gestion de dépendances au format CUDF

```

package: firefox
version: 2
provides: web
conflicts: firefox=3
installsize: 63000

package: firefox
version: 3
provides: web
conflicts: firefox=2
installsize: 68000

package: chrome
version: 10
provides: web
installsize: 50000

package: opera
version: 11
provides: web
installsize: 55000

package: thunderbird
version: 2
provides: mail
conflicts: thunderbird=3
installsize: 22000

package: thunderbird
version: 3
provides: mail
conflicts: thunderbird=2
installsize: 23000

package: evolution
version: 3
provides: mail
depends: gnome=3
installsize: 25000

package: lightning
version: 1
provides: calendar
installsize: 1000

package: gnome
version: 3
installsize: 230000

package: web-machine
version: 1
depends: web, mail

package: web-machine
version: 2
depends: web,mail,calendar

request: installe une machine web
install: web-machine

```

L'outil **p2cudf** permet de résoudre des problèmes de gestion de dépendances entre logiciels exprimés dans le format CUDF. Il est basé sur une traduction pseudo-booléenne de ce problème.