INFO-F-302
Informatique Fondamentale
Exercices - Modélisation en SAT *

Prof. Emmanuel Filiot

Exercice 1 – Modus Ponens/Modus Tolens

Question 1 On considère que la phrase suivante est vraie : "Si je suis belge, je suis européen". Quelles phrases constituent un raisonnement correct à partir de cette hypothèse ?

1. Mr A est belge, donc il est européen
2. Mme B est européenne, donc elle est belge
3. Mr C n'est pas belge, donc il n'est pas européen
4. Mme D n'est pas européenne, donc elle n'est pas belge

Exprimer "Si je suis belge, je suis européen" comme une formule "OU".

Question 2 On a un jeu de cartes telles que sur une face de chaque carte, il y a une lettre, et sur l'autre, un chiffre. Soient quatres cartes, dont on ne voit qu'une face :

1. une carte marquée d'un K
2. une carte marquée d'un A
3. une carte marquée d'un 5
4. une carte marquée d'un 8

On veut vérifier la phrase suivante : "Si une carte a une consonne sur une face, elle a un nombre impair sur l'autre".

— Exprimer cette phrase en formule $\rightarrow$ avec les variable x_c "il y a une consonne sur la carte" et x_p "il y a un nombre pair sur la carte"

— Quelles cartes doit-on retourner pour vérifier cette propriété pour les quatres cartes plus haut ?

— Ces réponses restent-elles valides si on ne sait pas que sur une face de chaque carte, il y a une lettre, et sur l'autre, un chiffre ? Quelles cartes doivent être retournées dans ce cas ?

Exercice 2 – Jeu 123Path Dans ce jeu, on va travailler sur une grille de départ de dimension $N \times N$ donnée par une fonction $G : \{1, \ldots, N\} \times \{1, \ldots, N\} \rightarrow \{1, \ldots, N^2\} \cup \{-\}$. L'objectif est de remplir cette grille avec les nombres de 1 à N^2 manquants, en respectant les contraintes suivantes : (1) chaque nombre n'apparaît qu'une seule fois ; (2) pour chaque nombre $x < N^2$, le nombre suivant (i.e. $x + 1$) est dans son voisinage direct (les voisins en diagonal sont aussi pris en compte) ; (3) le nombre N^2 doit être voisin du nombre 1.

Voici un exemple d'une grille 4×4 et une solution.

*http://www.ulb.ac.be/di/info-f-302/

① 1) B = belge , E = européen

$B \rightarrow E$

$\neg B \vee E$

1. Correct (modus ponens)

2. incorrect (une B peut être autre nationalité)

3. $\neg B \rightarrow \neg E$ (incorrect, peut être d'une autre nationalité)

4. $\neg E \rightarrow \neg B$ (correct, modus Tollens)

2) - $\alpha c \rightarrow \neg \alpha p$

$\neg \alpha c \rightarrow \neg \alpha p$

- [K] : on doit retourner pour vérifier que chiffre impair
  [A] : respecte la règle
  [5] : " " "
  [8] : vérifier que la lettre est une consonne
- Il faut retourner K, A et 8

-	-	-	-	15	11	10	9
-	-	12	-	16	14	12	8
1	-	-	-	1	13	7	6
-	-	4	-	2	3	4	5

Question 1 Etant donné N, exprimer par une formule en FNC les contraintes 2 et 3 (vous pouvez utiliser l'expression $x \bmod N^2$).

Question 2 Exprimer par une formule en FNC la contrainte 1.

Question 3 Comment tester l'unicité d'une solution ?

Question 4 Comment générer une deuxième solution ?

Exercice 3 – Modélisation : Gestion de Paquets Dans ce problème, il vous est demandé de justifier soigneusement vos réponses et d'expliquer votre modélisation.

Un paquet logiciel est un ensemble de fichiers contenant plusieurs données nécessaires à l'installation, la désinstallation et au fonctionnement d'un logiciel (code source, version, dépendances,...).

Un gestionnaire de paquets est un logiciel permettant d'installer, de désinstaller, et de mettre à jour des paquets logiciels installés sur un système informatique (par exemple, sur un système d'exploitation Linux ou OSX). Une des difficultés d'un gestionnaire de paquets est de gérer les problèmes de dépendances et de conflits entre paquets, ainsi que la multitude des versions disponibles pour chaque logiciel. Le nombre de paquets disponibles pour un système et le nombre de versions pouvant être très grands, résoudre les dépendances et les conflits posent des contraintes d'efficacité importantes.

Par exemple, on veut installer le paquet a version 1, qui dépend du paquet c version 1, mais aussi du paquet b, version 1 ou 2. Pour installer le paquet c version 1, on a besoin du paquet d version 1 ou 2. Cependant, le paquet a est en conflit avec le paquet d version 2, on ne peut donc pas les installer en même temps. Le paquet b, version 1 ou 2, ne dépend d'aucun autre paquet. Une solution est d'installer le paquet d version 1, puis le paquet c version 1, le paquet b version 1, puis enfin le paquet a version 1.

Chaque paquet contient ses dépendances et ses conflits. Dans notre exemple, on obtient :

```
paquet:   a(1)              paquet:  b(1)          paquet:  b(2)
depend:   b(1), b(2)        depend:                depend:
          c(1)              conflits:              conflits:
conflits: d(2)

paquet:  c(1)               paquet:  d(1)          paquet:  d(2)
depend:  d(1), d(2)         depend:                depend:
conflits:                   conflits:              conflits:
```

La première ligne indique le nom du paquet avec son numéro de version entre parenthèses, ensuite les dépendances, et enfin les conflits. Chaque ligne de dépendance se prend en conjonction et, au sein d'une ligne, on prend les dépendances en disjonction. Par exemple, $a(1)$ dépend du paquet $b(1)$ ou du paquet $b(2)$, ainsi que du paquet $c(1)$.

Formellement, on se donne un ensemble de paquets P, et deux fonctions $depend : P \to 2^{2^P}$ et $conflits : P \to 2^P$. Soit $p \in P$ un paquet. La donnée $depend(p)$ est un ensemble d'ensembles de paquets : si $depend(p) = \{dep_1, dots, dep_n\}$, où chaque dep_i est un ensemble de paquets, cela signifie que pour installer p, on devra installer, pour chaque $i \in \{1, \ldots, n\}$, au moins un paquet de dep_i. Enfin, $conflits(p)$ est l'ensemble des paquets avec lesquels p est en conflit.

Q. x[i][j] : i° ligne, j° colonne à la valeur k

0 : chaque case a au plus une valeur

$\bigwedge_{i,j}$

Dans notre exemple, on a donc $P = \{a(1), b(1), b(2), c(1), d(1), d(2)\}$ et les fonctions suivantes :

$p \in P$	$depend(p)$	$conflits(p)$
$a(1)$	$\{\{b(1), b(2)\}, \{c(1)\}\}$	$\{d(2)\}$
$b(1)$	$\varnothing$	$\varnothing$
$b(2)$	$\varnothing$	$\varnothing$
$c(1)$	$\{\{d(1), d(2)\}\}$	$\varnothing$
$d(1)$	$\varnothing$	$\varnothing$
$d(2)$	$\varnothing$	$\varnothing$

Supposons qu'on veuille installer un paquet p. Une *solution pour* p est un sous-ensemble de paquets $Q \subseteq P$ qu'on devra installer, et qui respecte les contraintes suivantes : (1) $p \in Q$ (on installe p) (2) pour tout paquet $q \in Q$, pour tout ensemble $dep \in depend(q)$, on a $dep \cap Q \neq \varnothing$. Autrement dit, pour tout $dep \in depend(q)$, il existe au moins un paquet $q' \in dep$ tel que $q' \in Q$ (q' sera donc installé) (3) pour tout paquet $q \in Q$ et tout $q' \in conflits(q)$, $q' \notin Q$. Autrement dit, aucun paquet $q' \in conflits(q)$ ne sera installé.

Question 1 Donner une solution pour $a(1)$ (sous la forme d'un ensemble) dans l'exemple précédent, ainsi qu'une solution pour f dans l'exemple suivant :

$p \in P$	$depend(p)$	$conflits(p)$
a	$\varnothing$	$\{a'\}$
a'	$\varnothing$	$\{a\}$
b	$\varnothing$	$\{b'\}$
b'	$\varnothing$	$\{b\}$
c	$\{\{a, b\}\}$	$\varnothing$
d	$\{\{a'\}\}$	$\varnothing$
f	$\{\{c\}, \{d\}\}$	$\varnothing$

Remarque Dans le dernier tableau, on a changé les notations pour les paquets (l'ensemble P est $\{a, a', b, b', c, d\}$). Cela ne change rien au problème, car un paquet est simplement représenté par un nom, qui peut être quelconque, et accompagné, ou non, d'un numéro de version.

Question 2 On vous demande maintenant de formuler le problème comme un problème SAT. Etant donnés un ensemble P de paquets, deux fonctions $depend$ et $conflits$, et un paquet $i \in P$ qu'on veut installer, expliquer comment construire une formule φ de la logique propositionnelle (en forme normale conjonctive) tel que φ est satisfaisable si et seulement si il existe une solution pour i. *Idée : utiliser P comme ensemble de propositions.*

Question 3 Le problème MAX-SAT-PARTIEL consiste, étant donnés deux ensembles de clauses A et M, à trouver une interprétation V des propositions qui va satisfaire toutes les clauses de A, et un maximum de clauses de M. Par exemple, pour les ensembles de clauses $A = \{\neg a \vee c, \neg c\}$ et $M = \{a \vee b, \neg a, \neg b \vee a\}$, on ne peut pas satisfaire toutes les clauses de $A \cup M$, mais en prenant $V(a) = V(b) = V(c) = 0$, on satisfait toutes les clauses de A et 2 clauses de M.

Etant donnés un ensemble P de paquets, deux fonctions $depend$ et $conflits$, et un paquet $i \in P$ qu'on veut installer, on voudrait trouver une solution pour i qui minimise le nombre de paquets à installer. Réduire ce problème en une instance du problème MAX-SAT-PARTIEL (utiliser votre réponse à la question précédente).

Question 4 Le problème MAX-POIDS-SAT-PARTIEL consiste, étant donnés deux ensembles de clauses A et M, et une fonction $poids : M \to \mathbb{Z}$, à trouver une interprétation V des propositions qui satisfait toutes les clauses de A, et maximiser la somme des poids des clauses satisfaites de M. Par exemple, pour les ensembles de clauses $A = \{\neg a \vee c, \neg c\}$ et $M = \{a \vee b, \neg a, \neg b \vee a\}$, avec la fonction $poids(a \vee b) = 5$, $poids(\neg a) = 1$ et $poids(\neg b \vee a) = 3$, en prenant $V(a) = V(c) = 0$ et $V(b) = 1$, on satisfait toutes les clauses de A et la clause $a \vee b$ de M, de poids 5. C'est le maximum possible. En effet, l'interprétation $V(a) = V(b) = V(c) = 0$ satisfait les deux clauses $\neg a$ et $\neg b \vee a$ de M, de somme totale 4.

Etant donnés un ensemble P de paquets, deux fonctions $depend$ et $conflits$, et un paquet $i \in P$ qu'on veut installer, et une fonction $taille : P \to \mathbb{N}$ qui donne, pour chaque paquet $p \in P$, la taille de ce paquet (en Mo), on voudrait trouver une solution pour i qui minimise la taille totale des paquets à télécharger. Réduire ce problème en une instance du problème MAX-POIDS-SAT-PARTIEL (inspirez-vous de votre réponse à la question précédente).

Question 5 Le problème MAX-POIDS$^+$-SAT-PARTIEL est identique au problème MAX-POIDS-SAT-PARTIEL, avec la restriction supplémentaire que la fonction $poids : M \to \mathbb{N}$ est positive.

Trouvez une réduction polynomiale de MAX-POIDS$^+$-SAT-PARTIEL à MAX-POIDS-SAT-PARTIEL, et inversement.

Question 6 On propose une réduction de MAX-POIDS$^+$-SAT-PARTIEL à MAX-SAT-PARTIEL : étant donnés une instance $(A, M, poids : M \to \mathbb{N})$ de MAX-POIDS$^+$-SAT-PARTIEL, on construit l'instance (A, M') de MAX-SAT-PARTIEL tel que pour tout $\varphi \in M$, φ apparait $poids(\varphi)$ fois dans M' (note : pour que M' reste un ensemble, on peut utiliser des formules évidemment équivalentes à φ, comme $\varphi \wedge \varphi$...). Par exemple, si $M = \{\neg a, b \vee c\}$, $poids(a) = 2$ et $poids(b \vee c) = 1$, alors $M' = \{a, a \wedge a, b \vee c\}$.

Est-ce une réduction du problème MAX-POIDS$^+$-SAT-PARTIEL à MAX-SAT-PARTIEL ? Est-elle polynomiale ?

Question 7 Etant donné un ensemble de clauses A, expliquer comment transformer (en temps polynomial), cet ensemble de clauses en une instance $(P, depend, conflits, i)$ où $i \in P$ est le paquet à installer, tel qu'il existe une solution pour i si et seulement si A est satisfaisable.