ENSTA ParisTech
2018-2019

Résolution efficace des modèles logiques – IA303

Travaux Dirigés & Travaux Pratiques

Feuille n° 2

Objectif(s)

- ★ Mise en œuvre de l'algorithme DPLL en version simple

Exercice 1 – Algorithme DPLL

Un encodage des PL formule en CNF est donné par le format DIMACS CNF. Un exemple de fichier est :

```
c  simple_v3_c2.cnf
c
p cnf 3 2
1 -3 0
2 3 -1 0
```

Toutes les lignes commençant par un caractère "c" sont des commentaires. Le contenu du fichier commence par les mots « p cnf », suivis du nombre de variables n et du nombre de clauses c du problème. Dans un fichier DIMACS CNF, une variable est représentée par un entier compris entre 1 et n. La négation : est représentée par le signe $-$. Une clause est représentée comme une liste de littéraux, séparés par des espaces, et terminée par un 0. Un problème est représenté comme une succession de clauses. Par exemple, le fichier DIMACS CNF donné en exemple encode la formule

$$(x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee \neg x_1)$$

On ne traitera pas la lecture d'un fichier DIMACS CNF mais on s'inspire de l'encodage des PL formule en CNF.

L'objectif de cet exercice est de mettre en œuvre une version simple de l'algorithme DPLL.

Question 1

Donnez une strucure de données qui représente une PL formule en CNF suivant le format DIMACS CNF.

Question 2

En vous inspirant du pseudo-code donné dans le cours mettez en œuvre l'algorithme DPLL simple.

Remarque : la bibliothèque SymPy de Python fourni une implémentation de l'algorithme DPLL qui pourra vous être utile pour vérifier votre implémentation de DPLL.

Par exemple, le code source Python utlise cet algorithme dans la fonction satisfiable

```
from sympy.logic.boolalg import And, Or, Implies, Equivalent, Not, to_cnf
from sympy.abc import p, q, r
from sympy.logic.inference import satisfiable

expr = Implies(p, Equivalent (q, r))
print(expr)

expr_cnf = to_cnf(expr)
print(expr_cnf)

print(satisfiable(expr_cnf))

from sympy.logic.utilities.dimacs import load
expr2_cnf = load('1 2 \n 3')
print(expr2_cnf)
print(satisfiable(expr2_cnf))
```