Laborator suplimentar

Laborator suplimentar

TODO

- Operatori în Prolog.
- Exerciții
  - definirea limbajului calculului propozițional clasic
  - prelucrarea formulelor

Scopul acestor exerciții este de a face prelucrări în vederea implementării unui SAT solver bazat pe rezoluție.

Operatori în Prolog

Operatori în Prolog

- Ați întâlnit până acum mai mulți operatori în Prolog: +, *, is, ...
- Fiecare operator are o precedență și o regulă pentru asociativitate.

Exemplu

```
?- X is 2+3+4.
X = 9.
true

?- 2+3+4=2+(3+4).
false.

?- 2+3+4=(2+3)+4.
true.
```

Operatori în Prolog

- Putem afla informații despre un operator folosind predicatul `current_op`

Exemplu

```
?- current_op(Precedence, Associativity, is).
Precedence = 700,
Associativity = xfx.

?-current_op(Precedence, Associativity, +).
Precedence = 200,
Associativity = fy
Precedence = 500,
Associativity = yfx.
```

- Observăm că operațiile cu precedență mai mică se efectuează primele.
- Ce înseamnă `xfx` și `fy`?

Operatori în Prolog

- Asociativitatea operatorilor este desemnată prin:

  `xf`, `yf`, `xfx`, `xfy`, `yfx`, `fy` or `fx`

  - `f` este functorul
  - `y` este un termen cu precedența mai mică sau egală cu a functorului
  - `x` este un termen cu precedența strict mai mică decât a functorului

Exemplu

Operatorul - binar are precedenta 500 și asociativitatea `yfy`. Verficați aceasta folosind `current_op` și înțelegeți exemplele de mai jos.

```
?- current_op(500, yfx, -).
true.
?- 10-5-2 = 10-(5-2).
false.
?- 10-5-2 = (10-5)-2.
true.
```

Operatori în Prolog

Pattern	**Associativity**		**Examples**
`yfx`	infix	left-associative	`+, -, *`
`xfy`	infix	right-associative	`,` (for subgoals)
`xfx`	infix	non-associative	`=, is, <` (i.e., no nesting)
`yfy`	makes no sense, structuring would be impossible		
`fy`	prefix	associative	`-` (i.e., `- - 5` allowed)
`fx`	prefix	non-associative	`:-` (i.e., `:- :- goal` not allowed)
`yf`	postfix	associative	
`xf`	postfix	non-associative	

sursa tabelului

Operatori în Prolog

- În Prolog putem defini operatori noi astfel

```
:- op(Precedence,Type,Name).
```

Atenție! Definirea unui operator este sintactică, nu spune nimic despre semnificația sa, care trebuie definită separat.

Exemplu

```
:- op(500, xf, is_dead).

kill(marsellus,zed).
is_dead(X) :- kill(_,X).
```

Citiți mai multe despre operatori:
SWI-Prolog
Learn Prolog Now!

Exercitii

Practică

Exercițiul 1: definiți limbajul logicii propoziționale clasice în Prolog.

Începeți prin a defini:

- variabilele: `is_var(a).  is_var(b).`
- operatorii: `nu`, `si`, `sau`, `imp`

```
:- op(620, xfy, si).
:- op(610, fy, nu).
```

Exemplu:

```
?- X= a si nu b.
X = a si nu b.
```

Practică

Exercițiul 2: scrieți un predicat care sa întoarcu a `true` dacă argumentul este o formulă corectă.

Exemplu:

```
?- formula(nu nu a si b sau c).
true.
```

Exercițiul 3: scrieți un predicat `inloc_imp(X,Y)` cu următorul efect: `Y` este `X` în care toate implicatiile au fost scrise folosind `sau` și `nu`.

Amintiți-vă că $\varphi \to \psi \sim \neg\varphi \vee \psi$

Exemplu:

```
?- inloc_imp(a imp b imp c, Y).
Y = nu a sau nu b sau c
```

Forma NNF

Fie φ o formulă din calculul propozițional clasic care nu conține implicații.

- Formula φ este în forma NNF dacă negația este numai pe variabile.

  Exemple:
  $p \wedge \neg q$ este în formă NNF
  $\neg(p \vee q) \wedge r$ nu este în formă NNF

- Formula φ poate fi adusă la forma NNF folosind:
  - regulile De Morgan
  $$\neg(\varphi \vee \psi) \sim \neg\varphi \wedge \neg\psi,$$
  $$\neg(\varphi \wedge \psi) \sim \neg\varphi \vee \neg\psi,$$
  - principiului dublei negații
  $$\neg\neg\psi \sim \psi$$

Exercițiul 4: scrieți un predicat `is_nnf(X)` care să întoarcă `true` dacă `X` este în formă NNF; vom presupune ca `X` nu conține implicații.

Practică

Exercițiul 5: scrieți un predicat `nnf(X,Y)` astfel încât `Y` să fie `X` în formă NNF; vom presupune ca `X` nu conține implicații.

Exemplu:

```
?- nnf(nu nu a, Y).
Y = a .

?- nnf(a, Y).
Y = a .

?- nnf(a si nu nu b, Y).
Y = a si b .

?- nnf(nu (a sau b) si nu nu c, Y).
Y = (nu a si nu b)si c
```

FND si FNC

- Un literal este o variabilă sau negația unei variabile.
- O formă normală disjunctivă (FND) este o disjuncție de conjuncții de literali.
- O formă normală conjunctivă (FNC) este o conjuncție de disjuncții de literali.
- Dacă φ este o formulă în forma NNF atunci ea poate fi adusă la FNC sau FND folosind distributivitatea

$$\varphi \vee (\psi \wedge \chi) \sim (\varphi \vee \psi) \wedge (\varphi \vee \chi),$$
$$(\psi \wedge \chi) \vee \varphi \sim (\psi \vee \varphi) \wedge (\chi \vee \varphi)$$

FNC

Orice formulă poate fi adusa la FNC prin urmatoarele transformări:

1. înlocuirea implicațiilor și echivalențelor

$$\varphi \to \psi \sim \neg\varphi \vee \psi,$$
$$\varphi \leftrightarrow \psi \sim (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$$

2. regulile De Morgan

$$\neg(\varphi \vee \psi) \sim \neg\varphi \wedge \neg\psi,$$
$$\neg(\varphi \wedge \psi) \sim \neg\varphi \vee \neg\psi,$$

3. principiului dublei negații

$$\neg\neg\psi \sim \psi$$

4. distributivitatea

$$\varphi \vee (\psi \wedge \chi) \sim (\varphi \vee \psi) \wedge (\varphi \vee \chi),$$
$$(\psi \wedge \chi) \vee \varphi \sim (\psi \vee \varphi) \wedge (\chi \vee \varphi)$$

Exemple

1. Determinați FNC pentru formula
$(\neg p \to \neg q) \to (p \to q)$
$\sim \neg(\neg p \to \neg q) \vee (p \to q)$
$\sim \neg(p \vee \neg q) \vee (\neg p \vee q)$
$\sim (\neg p \wedge q) \vee (\neg p \vee q)$
$\sim (\neg p \vee \neg p \vee q) \wedge (q \vee \neg p \vee q)$

2. Determinați FNC pentru formula
$\neg((p \wedge q) \to q)$
$\sim \neg(\neg(p \wedge q) \vee q)$
$\sim p \wedge q \wedge \neg q$

Practică

Exercițiul 6: scrieți un predicat `cnf(X,Y)` astfel încât `Y` să fie `X` în formă cnf.

Exemplu:

```
?- cnf(a,Y).
Y = a .

?- cnf(a imp b,Y).
Y = nu a sau b .

?- cnf(a imp b imp c,Y).
Y = nu a sau nu b sau c .

?- cnf(a imp b imp (c si d),Y).
Y = (nu a sau nu b sau c)si(nu a sau nu b sau d)
```

Pe data viitoare!