Erasmus Mundus master course in Data Mining and Knowledge Management a european master

UPMC SORBONNE UNIVERSITÉS

Logic and Knowledge Representation

Assignment n° 2

YEAR 2015-2016
J.-G. Ganascia
M.-J. Lesot

Remark

- You should not need more than 4 hours to solve the exercises.
- The solutions must be submitted as two files on the moodle website
  - a file whose name is of the form yourLastName.pdf containing the solution to the logic part (exercices 1 to 4)
  - a file whose name is of the form yourLastName.pl containing the solution to the prolog part (exercices 5 and 6).
- The submission deadline is Tuesday December 22nd, 6 :00pm.

1 Predicate calculus

Exercise 1 – Formalisation

Using the predicate and symbol functions given below,

Predicates		Functions	
$A(x)$	x is English	$e(x)$	denotes x worst enemy
$H(x,y)$	x hates y	n	denotes Napoléon
$C(x,y)$	x knows y		

formalise the following assertions in predicate calculus

1. any English person hates someone
2. everyone knows someone he hates and someone he does not hate
3. Napoléon's worst enemy is English
4. any English person hates his worst enemy
5. anyone who knows his worst enemy does not hate him

Exercise 2 – Interpretation and semantic

Consider a domain $\mathcal{D}$, a binary predicate R and the predicate of non equality $\neq$. Consider the formula $F_1 = \exists x \forall y (R(x,y) \vee x \neq y)$.

1. How many models with one element, i.e. such that $card(\mathcal{D}) = 1$, does F have, and what are they ?
2. Answer the same question for models with two elements, such that $card(\mathcal{D}) = 2$.
3. If the domain is $\mathcal{D} = \mathbb{N}$ and R is the relation lower than or equal to, is F_1 valid ?
4. Answer the same questions 1 to 3 for the formula $F_2 = \exists x \forall y (R(x,y) \wedge x \neq y)$.

Exercise 3 – Semantic trees

Consider a language with two constants, a and b, two functions, f with arity 1 and g with arity 2, four predicates, P and S with arity 1, Q and R with arity 2 and the following formulas

$$F_1 = \forall x \ \neg S(f(x)) \qquad F_3 = R(a,b)$$
$$F_2 = \forall y \forall z \ (P(z) \supset (S(y) \vee Q(y,z))) \qquad F_4 = \forall x \forall y \ (R(x,y) \supset P(g(x,f(y))))$$

1. Identify the set of clauses to consider, writing the formulas in the normal conjunctive form and in the Skolem form.
2. Indicate the considered Herbrand universe.
3. Consider $G = \exists x \exists y \exists z\ Q(y, g(z, f(x)))$. Show $F_1, F_2, F_3, F_4 \models G$ using semantic trees.

Exercise 4 – Resolution

Consider two binary predicates R and S and three formulas

$$
\begin{aligned}
F_1 &= \exists x \forall y R(x, y) \\
F_2 &= \forall x (\forall y R(x, y) \supset \exists z S(x, z)) \\
F_3 &= \forall x \forall y \forall z (R(x, y) \supset (S(x, z) \supset S(y, z)))
\end{aligned}
$$

1. Write a prenex and Skolem form for these three formulas.
2. Consider $G = \forall x \exists y S(x, y)$. Using resolution, show that $F_1, F_2, F_3 \models G$.

2 Prolog programming

Exercise 5 – Basic matrices operations

We consider matrices represented as lists of their rows, the latter being in turn represented as lists. For instance

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \quad \text{is represented as the list} \quad \texttt{[[1,2],[3,4],[5,6]]}$$

1. Write a predicate `size(M, NbRows, NbCols)` that is satisfied if, `M` being list of lists, `NbRows` is its number of rows and `NbCols` its number of columns. The predicate must fail if the list `M` does not represent a matrix, i.e. if not all rows have the same number of columns.

   Example :
   ```
   ?- size([[1,2],[3,4],[5,6]], NbRows, NbCols).
        NbRows = 3, NbCols = 2 ;
        No
   ?- size([[1,2],[3,4,5]], NbRows, NbCols).
        No
   ```

2. Write a predicate `rowI(M, I, RI)` that is satisfied if, `M` being a list of lists representing a matrix and `I` a positive integer, `RI` is the list representing the `I`th row of `M`. The predicate must fail is `I` is too large.

   Example :
   ```
   ?- rowI([[1,2],[3,4],[5,6]], 2, RI).
        RI = [3,4] ;
        No
   ```

3. Write a predicate `columnJ(M, J, CJ)` that is satisfied if, `M` being a list of lists representing a matrix and `J` a positive integer, `CJ` is the list representing the `J`th column of `M`. The predicate must fail is `J` is too large.

   Example :
   ```
   ?- columnJ([[1,2],[3,4],[5,6]], 1, CJ).
        CJ = [1,3,5] ;
        No
   ```

4. Write a predicate `product(M1, M2, R)` that is satisfied if, `M1` and `M2` being two lists of lists representing matrices, `M` is the list of list representing their product.

   Example :
   ```
   ?- product([[1,2],[3,4],[5,6]], [[1,1,1],[1,1,1]], M).
        M = [[3, 3, 3], [7, 7, 7], [11, 11, 11]] ;
        No
   ```

5. Write a predicate `traceMatrix(M, T)` that is satisfied if, `M` being a list of lists representing a **square** matrix, `T` is its trace (i.e. the sum of its diagonal elements).

   ExAmple :
   ```
   ?- traceMatrix([[1, 2, 3], [4, 5, 6], [7, 8, 9]],R).
        R = 15 ;
        No
   ```

6. Write a predicate `diagonal(M, D)` that is satisfied if, `M` being a list of lists representing a **square** matrix, `D` is diagonal, i.e. the list of its diagonal elements.

   Example : `?- diagonal([[1, 2, 3], [4, 5, 6], [7, 8, 9]],D).`
   ```
   D = [1, 5, 9] ;
   No
   ```

7. Write a predicate `identity(N, IM)` that is satisfied if `N` being a positive integer, `IM` is the identity matrix of size $N \times N$.

   Example : `?- identity(3,I).`
   ```
   I = [[1, 0, 0], [0, 1, 0], [0, 0, 1]] ;
   No
   ```

Exercise 6 – Computation of the matrix transpose

The aim of this exercise is to write an efficient implementation of the matrix transpose operation.

1. Write a predicate `listFirst(M, LF)` that is satisfied if, `M` being a list of lists representing a matrix, `LF` is the list of the first elements of each sublist.

   Example : `?- listFirst([[1,2,8],[3,4],[5,6]],LF).`
   ```
   LF = [1, 3, 5] ;
   No
   ```

2. Write a predicate `listFollowers(M, LF)` that is satisfied if, `M` being a list of lists representing a matrix, `LF` is the list of lists made of the sublists of `M` deprived of their first element.

   Example : `?- listFollowers([[1,2,8],[3,4],[5,6]],LF).`
   ```
   LF = [[2, 8], [4], [6]] ;
   No
   ```

3. Write a predicate `decompose(M, L1? L2)` that is satisfied if, `M` being a list of lists representing a matrix, `L1` is the list of the first elements of each sublist and `L2` is the list containing the sublists of `M` deprived of their first element.

   Warning : an efficient implementation of this predicate does not use the previous predicates.

   Example : `?- decompose([[1,2,8],[3,4],[5,6]], L1, L2).`
   ```
   L1 = [1, 3, 5]
   L2 = [[2, 8], [4], [6]] ;
   No
   ```

4. Write a predicate `transpose(M, R)` that is satisfied if, `M` being a list of lists representing a matrix, `R` is the list of list representing its transpose.

   Note : it is not asked to verify that the list of lists `M` corresponds to a matrix, i.e. that each sublist contains the same number of elements.

   Example : `?- transpose([[1,2],[3,4],[5,6]], R).`
   ```
   R = [[1,3,5],[2,4,6]] ;
   No
   ```