Laborator 4

2018-2019

Programare Logică

Laboratorul 4

TODO

- Rezolvarea unui puzzle
- *Countdown* - cel mai lung cuvânt
- DCG - gramatici cu clauze definite

Practică

Exercițiul 1: Zebra puzzle

Zebra puzzle este unul dintre cele mai cunoscute puzzle-uri logice.

Citiți mai jos mai multe detalii despre acest puzzle:
`https://en.wikipedia.org/wiki/Zebra_Puzzle`

Vom rezolva următoarea versiune publicată în 1962 în care fiecare personaj

- locuiește într-o casă care are o anumită culoare,
- are o naționalitate,
- are un anumit animal de companie,
- are o băutură preferată,
- fumează un anumit tip de țigări.

Practică

Exercițiul 1 (cont.)

Vrem să aflăm cine are o zebră având următoarele informații:

1. Sunt cinci case.
2. Englezul locuiește în casa roșie.
3. Spaniolul are un câine.
4. În casa verde se bea cafea.
5. Ucraineanul bea ceai.
6. Casa verde este imediat în dreapta casei bej.
7. Fumătorul de "Old Gold" are melci.
8. În casa galbenă se fumează "Kools".

Practică

Exercițiul 1 (cont.)

9. În casa din mjloc se bea lapte.
10. Norvegianul locuiește în prima casă.
11. Fumătorul de "Chesterfields" locuiește lângă cel care are o vulpe.
12. "Kools" sunt fumate în casa de lângă cea în care se ține calul.
13. Fumătorul de "Lucky Strike" bea suc de portocale.
14. Japonezul fumează "Parliaments".
15. Norvegianul locuiește lângă casa albastră.

Practică

Exercițiul 1 (cont.)

Vom rezolva acest puzzle în Prolog.

1) Definiți un predicat `la_dreapta(X,Y)` care este adevărat când numerele X și Y sunt consecutive, X fiind cel mai mare dintre ele.

2) Definiți un predicat `la_stanga(X,Y)` care este adevărat când numerele X și Y sunt consecutive, Y fiind cel mai mare dintre ele.

3) Definiți un predicat `langa(X,Y)` care este adevărat când numerele X și Y sunt consecutive.

Practică

Exercițiul 1 (cont.)

În continuare definim un predicat `solutie(Strada,PosesorZebra)` în care includem toate informațiile pe care le deținem:

```
solutie(Strada,PosesorZebra) :-
  Strada = [
      casa(1,_,_,_,_,_,_),
      casa(2,_,_,_,_,_,_),
      casa(3,_,_,_,_,_,_),
      casa(4,_,_,_,_,_,_),
      casa(5,_,_,_,_,_,_)],
    member(casa(_,englez,rosie,_,_,_), Strada),
    member(casa(_,spaniol,_,caine,_,_), Strada),
    member(casa(_,_,verde,_,cafea,_), Strada),
    ...,
    member(casa(_,PosesorZebra,_,zebra,_,_), Strada),
```

Completați toate informațiile pentru a afla soluția ținând cont de codarea:

```
casa(Numar,Nationalitate,Culoare,AnimalCompanie,Bautura,Tigari)
```

Practică

Exercițiul 2: Cel mai lung cuvânt

Acest exemplu este din
Ulle Endriss, *Lecture Notes – An Introduction to Prolog Programming.*

Countdown este un joc de televiziune popular în Marea Britanie în care jucătorii trebuie să găsească un cuvânt cât mai lung cu literele dintr-o mulțime dată de nouă litere.

Să încercăm să rezolvăm acest joc cu Prolog!

Practică

Exercițiul 2: Cel mai lung cuvânt

Concret, vom încerca să găsim o soluție optimă pentru următorul joc:

Primind o listă cu litere din alfabet (nu neapărat unice),
trebuie să construim cel mai lung cuvânt format din literele date
(pot rămâne litere nefolosite).

Vom rezolva jocul pentru cuvinte din limba engleză.

Scorul obținut este lungimea cuvântului găsit.

Practică

Exercițiul 2 (cont.)

Scopul final este de a construi un predicat în Prolog `topsolution/2`:

dându-se o listă de litere în primul său argument, trebuie să returneze în al doilea argument o soluție cât mai bună, adică un cuvânt din limba engleză de lungime maximă care poate fi format cu literele din primul argument.

```
?- topsolution([r,d,i,o,m,t,a,p,v],Word).
Word = dioptra
```

Practică

Exercițiul 2 (cont.)

Începeți prin a descărca fișierul `words.pl` în același director cu fișierul programului vostru.

Acest fișier conține o listă cu peste 350.000 de cuvinte din limba engleză, de la `a` la `zyzzyva`, sub formă de fapte.

Includeți linia `:- include('words.pl').` în programul vostru pentru a putea folosi aceste fapte.

Practică

Exercițiul 2 (cont.)

Predicatul predefinit în Prolog `atom_chars(Atom,CharList)` descompune un atom într-o listă de caractere.

Folosiți acest predicat pentru a defini un predicat `word_letters/2` care transformă un cuvânt (i.e, un atom în Prolog) într-o listă de litere.
De exemplu:

```
?- word_letters(hello,X).
X = [h,e,l,l,o]
```

Ca o paranteza, observați că puteți folosi acest predicat pentru a găsi cuvinte în engleză de 45 de litere:

```
?- word(Word), word_letters(Word,Letters),
   length(Letters,45).
```

Practică

Exercițiul 2 (cont.)

Mai departe, scrieți un predicat `cover/2` care, primind două liste, verifică dacă a doua listă "acoperă" prima listă (i.e., verifică dacă fiecare element care apare de k ori în prima listă apare de cel puțin k ori în a doua listă).

De exemplu

```
?- cover([a,e,i,o], [m,o,n,k,e,y,b,r,a,i,n]).
true

?- cover([e,e,l], [h,e,l,l,o]).
false
```

Practică

Exercițiul 2 (cont.)

Scrieți un predicat `solution/3` care primind o listă de litere ca prim argument și un scor dorit ca al treilea argument, returnează prin al doilea argument un cuvânt cu lungimea egală cu scorul dorit, "acoperit" de lista respectivă de litere.

De exemplu

```
?- solution([g,i,g,c,n,o,a,s,t], Word, 3).
Word = act
```

Practică

Exercițiul 2 (cont.)

Acum implementați predicatul `topsolution/3`.

Testați, de exemplu, predicatul definit pe mulțimea de litere:

```
[y,c,a,l,b,e,o,s,x]
```

Aceasta este una listele de litere folosite în ediția de *Countdown* din 18 Decembrie 2002 din Marea Britanie în care Julian Fell a obținut cel mai mare scor din istoria concursului. Pentru lista de mai sus, el a găsit cuvântul *cables*, câștigând astfel 6 puncte.

Poate programul vostru să bată acest scor?

Gramatici cu clauze definite

Înainte de a trece mai departe, amintiti-vă din *Cursul 5* reprezentarea unei gramatici independente de context în Prolog.

În fișierul `dcg.pl` găsiți exemplele discutate la curs.

Exercițiu

- Puneți întrebări, testând gramaticile definite la curs.

```
?- phrase(s,SenL), atomic_list_concat(SenL,' ',Sen). ?-
phrase(nat, Num), atom_chars(Natural,Num).
```

- Folosiți predicatul `listing` pentru a vedea modul în care notația DCG `-->` este implementată automat.

```
?- listing(s).
?- listig(nat).
```

Practică

Exercițiul 3: numere întregi

- Folosind notația DCG, definiți numerele întregi ca șiruri de forma

  $$w\mathtt{o}$$

  unde w este un cuvânt format cu literele `s` și `p`. Exemple de numere întregi generate de gramatică:

  `o`, `so`, `spo`, `po`, `pspspppo`, etc

- Scrieți o funcție care convertește numarul generat de gramatică în valoarea corespunzătoare. Puteți lucra direct cu numerele reprezentate ca liste.

```
?- value([s,p,s,p,p,o], V).
V = -1.
```

Practică

Exercițiul 4: limba română

Gramatica definită la curs generează fraze de tipul "*A boy loves a girl*".

Scrieți o gramatică care va genera(recunoaște) fraze asemănătoare în limba română, tinând cont de genul substantivului(considerați numai articolele nehotărâte):

```
?- phrase(sr, [o, fata, iubeste, un, baiat] ).
true
```

Mai mult despre DCG

Atunci când programăm cu clauze definite putem:

- să adăugăm predicate în corpul clauzelor; aceste predicate trebuie incluse între acolade. De exemplu, dacă avem un predicat `lex/2` care precizează pentru fiecare cuvânt categoria gramaticală

  ```
  lex(a,art).    lex(the,art).
  ```

  putem rescrie regulile astfel:

  ```
  det --> {lex(Word,art)},[Word].
  ```

- să adăugăm parametrii în definițiile corespunzătoare neterminalelor:

  ```
  lex(o,art,fem).
  lex(un,art,masc).
  ```

  putem rescrie regulile astfel:

  ```
  art(G) --> {lex(Word,art,G)},[Word].
  ```

Puteți verifica cu `listing` implementarea automată a definițiilor.

Practică

Exercițiul 5

În acest exercițiu vom folosi facilitățile DCG din slide-ul anterior.

- Folosind notația DCG, definiți o gramatică care recunoaște numai numerele naturale (în notația `[s,s,o]`) mai mici decât o valoare maxima fixată.

```
?- phrase(nat_max(3),[s,s,s,o]).
true
?- phrase(nat_max(3),[s,s,s,s,o]).
false.
```

- Definiți o bază de cunoștințe lexicale pentru exemplul din Exercițiul 4, precizând categoria gramaticală și genul (unde este cazul). Rescrieți gramatica într-o formă cât mai compactă.

Pe data viitoare!