PROLOG lab 1

Po zakończeniu laboratorium wszystkie zrobione zadania należy przesłać pocztą elektroniczną do prowadzącego zajęcia!

1. Prolog - implementacja

Podczas zajęć wykorzystywana będzie darmowa wersja interpretera Prologu **SWI-Prolog** do pobrania ze strony www.swi-prolog.org

2. Wprowadzenie

Prolog jest językiem programowania w logice (metoda programowania, w której program jest pewnym zestawem zależności, natomiast obliczenia są dowodem pewnego twierdzenia, opartym o podane zależności), w przeciwieństwie do popularnych proceduralnych języków programowania (np. C, PASCAL, Java). Prolog jest językiem deklaratywnym, co oznacza, że pisząc program skupiamy się na definicjach obiektów i związkach między nimi, a nie na sposobie wykonania danego zadania. Programowanie deklaratywne polega na wyspecyfikowaniu co jest problemem do rozwiązania, a nie jak go rozwiązać, czyli specyfikujemy cel i nie interesujemy się algorytmem jego rozwiązania. Podejście to różni się od przypadku programowania proceduralnego, gdzie musimy podać algorytm rozwiązujący pewien problem.

Programowanie w Prologu polega na:

- zdefiniowaniu zbioru *faktów* dotyczących obiektów i związków między nimi
- zdefiniowaniu *reguł* dotyczących obiektów i związków między nimi
- zadawania *zapytań* o obiekty i związki między nimi

Prolog stosuje się tam, gdzie wykorzystuje się metody sztucznej inteligencji oraz potrzebna jest symboliczna reprezentacja wiedzy np.: w przetwarzaniu języka naturalnego, w systemach eksperckich, w automatycznym dowodzeniu twierdzeń, w automatyce, w grach komputerowych etc.

3. Podstawowe definicje

Program napisany w Prologu jest zbiorem faktów i/lub reguł nazywanych *klauzulami*.

3.1 Fakty

Definiowane są jako zawsze prawdziwe związki między obiektami, np.:

lubi(maria, wino).

kobieta(maria).

rodzic(roman, maria).

W powyższym przykładzie zostały zapisane fakty typu: „Maria lubi wino", „Maria jest kobietą" oraz „Roman jest rodzicem Marii". Przy czym „lubi", „kobieta", „rodzic" są **predykatami**. Należy pamiętać o kropkach kończących deklaracje każdego faktu.

Fakty mogą też być zapisane za pomocą bardziej skomplikowanych struktur:

ksiazka(tytul('Dziady'), autor(adam, mickiewicz)).

3.2 Zapytania

W Prologu istnieje możliwość formułowania pytań dotyczących zdefiniowanych relacji.

Przykładowe zapytanie: „Czy Maria lubi wino?"

Zapis w Prologu:

?-lubi(maria, wino).

Odpowiedź:

Yes

Inne zapytanie:

?-lubi(kasia, wino).

Odpowiedź:

No

Baza znanych faktów zostaje przeszukana w takiej kolejności w jakiej fakty zostały wprowadzone i jeśli napotkany zostanie fakt, o który pytamy, odpowiedź będzie „tak", a zatem prawdziwość faktu zostanie potwierdzona. Jeśli fakt nie zostanie znaleziony, Prolog odpowiada „nie". Odpowiedź „nie" w Prologu nie oznacza, ze coś jest nieprawdziwe, ale że Prolog na podstawie znanych mu faktów (i reguł) nie jest w stanie tego potwierdzić.

3.3 Stałe i zmienne

W prologu stałe pisane są z małej litery (np. ***maria, wino***). Zmienne natomiast pisane są z wielkiej litery lub zaczynają się od znaku podkreślenia, np.

Y

_ktolubi

Ktolubi

Zmienne w Prologu można wykorzystać do formułowania pytań szczegółowych:

„Kto jest rodzicem Marii?"

Zapis w Prologu:

?-rodzic(X, maria).

Odpowiedź:

X=roman

Odpowiedzi na pytanie szczegółowe może być kilka. Naciśnięcie **ENTER s**powoduje zakończenie poszukiwań kolejnych rozwiązań. Naciśniecie średnika „ ; ” powoduje poszukiwanie alternatywnych rozwiązań (średnik jest w Prologu oznaczeniem logicznej operacji **OR)**:

X=roman;

No - nie ma więcej odpowiedzi

Pytanie szczegółowe może mieć węższy lub szerszy zakres np. „Dla jakich X i Y, X jest rodzicem Y?”

Zapis w Prologu:

?-rodzic(X,Y).

Do zmiennych przypisywane są kolejne znalezione wartości (**uzgadnianie**), które pasują do zapytania. Odbywa się to w procesie zwanym **nawracaniem.**

Zmienna posiada zasięg tylko w ramach danej klauzuli.

Zmienne anonimowe oznaczane podkreśleniem „_” używane są wtedy gdy nie interesuje nas wartość zmiennej (nie zostanie ona wyświetlona w wynikach zapytania), np.:

„Podaj wszystkich rodziców”

Zapis w Prologu:

?-rodzic(Y,_).

3.4 Komentarze

- blokowe /**/
- do końca linii %

3.5 Reguły

Reguła składa się głowy reguły oraz ciała oddzielonych znakiem „:-”. Głowa reguły składa się tylko z jednego predykatu, natomiast ciało reguły może być koniunkcją dowolnej liczby warunków (oddzielonych przecinkami – w Prologu oznaczają one logiczne **AND**). Aby spełniona była przesłanka reguły, spełnione muszą być wszystkie jej podcele.

Zapiszmy w Prologu regułę, że *maria* lubi *coś*, jeśli tylko to *cos* jest *tanie* i *zdrowe*.

lubi(maria, X) :- tanie(X), zdrowe(X).

3.6 Arytmetyka

Prolog umożliwia operacje na liczbach, pamiętać jednak trzeba o różnicach w zapisie:

*X = 3*5-1.*

a zapisem

*X is 3*5-1.*

Sprawdź wynik zapytania:

*X=3*5-1, display(X).*

Więcej informacji na temat możliwości jakie daje SWI-Prolog w zakresie działań arytmetycznych na stronie: http://gollem.science.uva.nl/SWI-Prolog/Manual/arith.html

3.7 Odcięcie

Odcięcie jest wbudowanym predykatem oznaczanym poprzez „!" i służy do kontroli nawracania. Jest zawsze spełniony i zwiększa efektywność poprzez jawne informowanie Prologa, żeby nie rozważał alternatyw, co do których wiemy, że nie będą spełnione.

Przykładowe zadanie:

Oblicz sumę dwóch liczb, jeśli pierwsza z nich jest mniejsza od 6, w przeciwnym przypadku wynik powinien mieć taką samą wartość jak druga liczba.

suma(X,Y,Z):-X<6, Z is X+Y.

suma(X,Y,Z):-Z is Y.

Oczywiste jest, że jeśli nie jest spełniony warunek w pierwszej klauzuli predykatu *suma*, to wykorzystana jest druga reguła. W takim przypadku Prolog prawidłowo odpowie na pytanie:

suma(2,3,Res).

Problem pojawi się natomiast gdy zadamy pytanie:

suma(7,5,Res).

Pierwsza odpowiedz będzie prawidłowa, jeśli jednak naciśniemy średnik, dostaniemy drugą odpowiedź, wywnioskowaną z drugiej reguły. Aby uniknąć takich sytuacji stosuje się mechanizm **odcięcia.**

suma(X,Y,Z):-X<6, Z is X+Y, !.

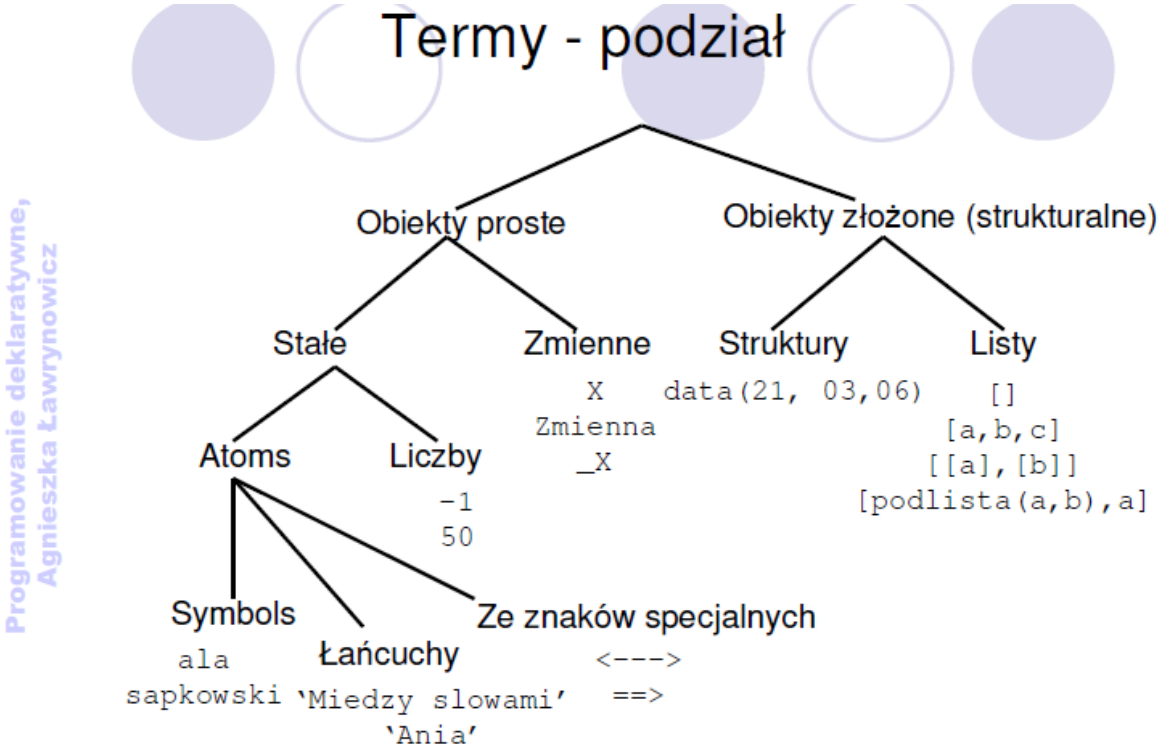
suma(X,Y,Z):-Z is Y.

Odcięcie spowoduje, że po dojściu do danego miejsca, Prolog nie będzie próbował uzgadniać ponownie celów po lewej stronie znaku „!".

3.7 Termy

Programy w Prologu budowane są na bazie termów. Wyróżniamy trzy ich rodzaje:

- stałe
- zmienne
- termy złożone



Obiekty złożone składają się z funktora i argumentów, przy czym argumenty same mogą być złożone np. *ma(jan, ksiazka(tytul(ulisses),autor(joyce)))*. Struktury wykorzystywane są do grupowania danych.

3.8 Unifikacja (dopasowanie)

Operator dopasowania oznaczany jest jako „=”. Mówimy, że dwa termy można zunifikować, gdy można je do siebie dopasować. Dopasowanie dwóch termów X i Y zachodzi gdy:

- jeśli X jest zmienną a Y dowolnym termem. W takiej sytuacji pod X podpisywana jest wartość Y, np.

  ?-jedzie(jan, auto) = X.
- obydwa termy są identycznymi stałymi (liczby całkowite i atomy zawsze są sobie równe), np.

  papier = papier

  1203 = 1203
- X i Y są termami złożonymi i jednocześnie mają taką samą liczbę argumentów, ten sam symbol funkcyjny oraz zachodzi dopasowanie pomiędzy wszystkimi argumentami obu termów.

Porównywanie termów: http://gollem.science.uva.nl/SWI-Prolog/Manual/compare.html.

4. Literatura i materiały

„Prolog. Programowanie” , W.F. Clocksin, C.S. Mellish, HELION

„Prolog, język sztucznej inteligencji”, Eugeniusz Gatnar, Katarzyna Stąpor, Wyd. PLJ

Internet:

po polsku:

http://www.im.pwr.wroc.pl/~przemko/prolog/

http://free.of.pl/p/prolog/strona2/prolog.php

po angielsku:

http://www.compapp.dcu.ie/~alex/LOGIC/start.html

http://kti.ms.mff.cuni.cz/~bartak/prolog/

http://www.amzi.com/AdventureInProlog/

http://computing.unn.ac.uk/staff/cgpb4/prologbook/book.html

http://appsrv.cse.cuhk.edu.hk/~csc4510/prolog/tutorial.1/1.htm

http://cs.wwc.edu/KU/PR/Prolog.html#extralogical

http://www.csupomona.edu/~jrfisher/www/prolog_tutorial/contents.html

http://www.cs.may.ie/~jpower/Courses/PROLOG/

manual SWI-Prolog

http://www.swi-prolog.org/documentation.html