Programowanie w Logice

Przykłady programów

Przemysław Kobylański

Przykłady programów

Interpreter prostego języka imperatywnego

- Język **Imperator**[1] jest prostym językiem imperatywnym.
- Jego składnię opisuje poniższa gramatyka BNF:

```
PROGRAM ::=
PROGRAM ::= INSTRUKCJA ; PROGRAM

INSTRUKCJA ::= IDENTYFIKATOR := WYRAŻENIE
INSTRUKCJA ::= read IDENTYFIKATOR
INSTRUKCJA ::= write WYRAŻENIE
INSTRUKCJA ::= if WARUNEK then PROGRAM fi
INSTRUKCJA ::= if WARUNEK then PROGRAM else PROGRAM fi
INSTRUKCJA ::= while WARUNEK do PROGRAM od
```

[1] Nazwę zaproponował MKI.

Przykłady programów

Interpreter prostego języka imperatywnego

```
WYRAŻENIE ::= SKŁADNIK + WYRAŻENIE
WYRAŻENIE ::= SKŁADNIK - WYRAŻENIE
WYRAŻENIE ::= SKŁADNIK

 SKŁADNIK ::= CZYNNIK * SKŁADNIK
 SKŁADNIK ::= CZYNNIK / SKŁADNIK
 SKŁADNIK ::= CZYNNIK mod SKŁADNIK
 SKŁADNIK ::= CZYNNIK

  CZYNNIK ::= IDENTYFIKATOR
  CZYNNIK ::= LICZBA_NATURALNA
  CZYNNIK ::= ( WYRAŻENIE )
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
    WARUNEK ::= KONIUNKCJA or WARUNEK
    WARUNEK ::= KONIUNKCJA

KONIUNKCJA ::= PROSTY and KONIUNKCJA
KONIUNKCJA ::= PROSTY

     PROSTY ::= WYRAŻENIE = WYRAŻENIE
     PROSTY ::= WYRAŻENIE /= WYRAŻENIE
     PROSTY ::= WYRAŻENIE < WYRAŻENIE
     PROSTY ::= WYRAŻENIE > WYRAŻENIE
     PROSTY ::= WYRAŻENIE >= WYRAŻENIE
     PROSTY ::= WYRAŻENIE =< WYRAŻENIE
     PROSTY ::= ( WARUNEK )
```

Przykłady programów

Interpreter prostego języka imperatywnego

Example (Obliczenie sumy liczb A i B)

```
read A;
read B;
X := A;
Y := B;
while Y > 0 do
  X := X + 1;
  Y := Y - 1;
od;
write X;
```

Przykłady programów

Interpreter prostego języka imperatywnego

- Program reprezentowany jest listą złożoną z termów.
- Każdy term reprezentuje jedną instrukcję.
- Jeśli instrukcja jest złożona, to term zawiera jako podterm listę termów reprezentujących zagnieżdżone instrukcje.

```
PROGRAM = [ ]
PROGRAM = [INSTRUKCJA | PROGRAM]

INSTRUKCJA = assign(ID, WYRAŻENIE)
INSTRUKCJA = read(ID)
INSTRUKCJA = write(WYRAŻENIE)
INSTRUKCJA = if(WARUNEK, PROGRAM)
INSTRUKCJA = if(WARUNEK, PROGRAM, PROGRAM)
INSTRUKCJA = while(WARUNEK, PROGRAM)
```

Przykłady programów

Interpreter prostego języka imperatywnego

Wyrażenie jest albo prostym wyrażeniem będącym zmienną lub liczbą naturalną albo wyrażeniem złożonym będącym sumą, różnicą, iloczynem lub ilorazem dwóch wyrażeń:

```
WYRAŻENIE = id(ID)
WYRAŻENIE = int(NUM)
WYRAŻENIE = WYRAŻENIE + WYRAŻENIE
WYRAŻENIE = WYRAŻENIE - WYRAŻENIE
WYRAŻENIE = WYRAŻENIE * WYRAŻENIE
WYRAŻENIE = WYRAŻENIE / WYRAŻENIE
WYRAŻENIE = WYRAŻENIE mod WYRAŻENIE
```

Przykłady programów

Interpreter prostego języka imperatywnego

Warunek jest relacją równości, różności, mniejszości, większości (słabej lub silnej) między wartościami dwóch wyrażeń albo alternatywą lub koniunkcją dwóch warunków:

```
WARUNEK = WYRAŻENIE =:= WYRAŻENIE
WARUNEK = WYRAŻENIE =\= WYRAŻENIE
WARUNEK = WYRAŻENIE < WYRAŻENIE
WARUNEK = WYRAŻENIE > WYRAŻENIE
WARUNEK = WYRAŻENIE =< WYRAŻENIE
WARUNEK = WYRAŻENIE >= WYRAŻENIE
WARUNEK = WARUNEK ; WARUNEK
WARUNEK = WARUNEK , WARUNEK
```

Przykłady programów

Interpreter prostego języka imperatywnego

Example (Sumowanie liczb od 1 do N)

```
program1([
    read('N'),
    assign('SUM', int(0)),
    while(id('N') > int(0),
          [assign('SUM', id('SUM') + id('N')),
           assign('N', id('N') - int(1))]),
    write(id('SUM'))]).
```

Przykłady programów

Interpreter prostego języka imperatywnego

Example (Obliczenie ilorazu D i reszty R z dzielenia M przez N)

```
program2([
    read('M'),
    read('N'),
    assign('D', int(0)),
    assign('R', id('M')),
    while(id('R') > id('N'),
          [assign('D', id('D') + int(1)),
           assign('R', id('R') - id('N'))]),
    write(id('D')),
    write(id('R'))]).
```

Przykłady programów

Interpreter prostego języka imperatywnego

- Bieżący stan obliczeń zapisywać będziemy w postaci listy asocjacji.
- Każda asocjacja jest termem w postaci `ID = Wartość`, gdzie `ID` jest nazwą zmiennej a `Wartość` jest bieżącą wartością tej zmiennej.

```
% podstaw(+Stare, +ID, +Wartosc, -Nowe)
podstaw([], ID, N, [ID = N]).
podstaw([ID=_ | AS], ID, N, [ID=N | AS]) :- !.
podstaw([ID1=W1 | AS1], ID, N, [ID1=W1 | AS2]) :-
        podstaw(AS1, ID, N, AS2).

% pobierz(+Asocjacje, +ID, -Wartosc)
pobierz([ID=N | _], ID, N) :- !.
pobierz([_ | AS], ID, N) :-
        pobierz(AS, ID, N).
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
% wartosc(+Wyrazenie, +Asocjacje, -Wartosc)
wartosc(int(N), _, N).
wartosc(id(ID), AS, N) :-
        pobierz(AS, ID, N).
wartosc(W1 + W2, AS, N) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N is N1 + N2.
wartosc(W1 - W2, AS, N) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N is N1 - N2.
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
wartosc(W1 * W2, AS, N) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N is N1 * N2.
wartosc(W1 / W2, AS, N) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N2 =\= 0, N is N1 div N2.
wartosc(W1 mod W2, AS, N) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N2 =\= 0,
        N is N1 mod N2.
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
% prawda(+Warunek, +Asocjacje)
prawda(W1 =:= W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 =:= N2.
prawda(W1 =\= W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 =\= N2.
prawda(W1 < W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 < N2.
prawda(W1 > W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 > N2.
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
prawda(W1 >= W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 >= N2.
prawda(W1 =< W2, AS) :-
        wartosc(W1, AS, N1), wartosc(W2, AS, N2),
        N1 =< N2.
prawda((W1, W2), AS) :-
        prawda(W1, AS),
        prawda(W2, AS).
prawda((W1; W2), AS) :-
        (   prawda(W1, AS),
            !
        ;   prawda(W2, AS)).
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
% interpreter(+Program, +Asocjacje)
interpreter([], _).
interpreter([read(ID) | PGM], AS) :- !,
        read(N),
        integer(N),
        podstaw(AS, ID, N, AS1),
        interpreter(PGM, AS1).
interpreter([write(W) | PGM], AS) :- !,
        wartosc(W, AS, WART),
        write(WART), nl,
        interpreter(PGM, AS).
interpreter([assign(ID, W) | PGM], AS) :- !,
        wartosc(W, AS, WAR),
        podstaw(AS, ID, WAR, AS1),
        interpreter(PGM, AS1).
```

Przykłady programów

Interpreter prostego języka imperatywnego

```
interpreter([if(C, P) | PGM], ASO) :- !,
        interpreter([if(C, P, []) | PGM], AS).
interpreter([if(C, P1, P2) | PGM], AS) :- !,
        (   prawda(C, AS)
        ->  append(P1, PGM, DALEJ)
        ;   append(P2, PGM, DALEJ)),
        interpreter(DALEJ, AS).
interpreter([while(C, P) | PGM], AS) :- !,
        append(P, [while(C, P)], DALEJ),
        interpreter([if(C, DALEJ) | PGM], AS).

% interpreter(+Program)
interpreter(PROGRAM) :-
        interpreter(PROGRAM, []).
```

Przykłady programów

Interpreter prostego języka imperatywnego

Example (Sumowanie)

```
?- program1(X), interpreter(X).
|: 10.
55
X = [read('N'), assign('SUM', int(0)),
     while(id('N')>int(0), [assign('SUM', id('SUM')
     +id('N')), assign('N', id('N')-int(1))]),
     write(id('SUM'))].
```

Przykłady programów

Interpreter prostego języka imperatywnego

Example (Iloraz i reszta)

```
?- program2(X), interpreter(X).
|: 123.
|: 13.
9
6
X = [read('M'), read('N'), assign('D', int(0)),
     assign('R', id('M')), while(id('R')>id('N'),
     [assign('D', id(...)+int(...)), assign('R', ...
      - ...)]), write(id('D')), write(id('R'))].

?- X is 9*13+6.
X = 123.
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

- Zakładamy, że predykaty zapisane są w postaci klauzul, których ciała zawierają tylko koniunkcje formuł atomowych (nie ma negacji, alternatyw i implikacji).
- Wszystkie predykaty są zdefiniowane przez nas (nie korzystamy z predykatów wbudowanych, które mogą być napisane np. w języku C).

Przykłady programów

Interpreter podzbioru Prologu w Prologu

Example (Konkatenacja list na liście)

```
app([], X, X).
app([X | L1], L2, [X | L3]) :-
        app(L1, L2, L3).

app([], []).
app([L1 | L2], L3) :-
        app(L1, L4, L3),
        app(L2, L4).
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

- Predykat `clause(H, B)` dostarcza głowę `H` i ciało `B` klauzuli.
- `H` jest unifikowane z głową a `B` z ciałem klauzuli.
- Fakt ma ciało równe `true`.

```
?- clause(app([], A, B), C).
A = B,
C = true.

?- clause(app(A, B, C), D).
A = [],
B = C,
D = true ;
A = [_G4888|_G4889],
C = [_G4888|_G4892],
D = app(_G4889, B, _G4892).
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

```
?- clause(app(A, B), C).
A = B, B = [],
C = true ;
A = [_G4870|_G4871],
C =  (app(_G4870, _G4874, B), app(_G4871, _G4874)).
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

- Interpreter Prologu w Prologu zapiszemy w postaci predykatu `udowodnij(Cel)`, gdzie `Cel` jest zadanym celem do udowodnienia.

```
udowodnij(true) :- !.
udowodnij((G1, G2)) :- !,
        udowodnij(G1),
        udowodnij(G2).
udowodnij(A) :-
        clause(A, B),
        udowodnij(B).
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

```
?- udowodnij(app(X, Y, [1, 2, 3])).
X = [],
Y = [1, 2, 3] ;
X = [1],
Y = [2, 3] ;
X = [1, 2],
Y = [3] ;
X = [1, 2, 3],
Y = [] ;
false.
```

Przykłady programów

Interpreter podzbioru Prologu w Prologu

```
?- udowodnij(app([1, 2, 3], X, [1, 2, 3, 4 | Y])).
X = [4|Y].

?- udowodnij(app([[1, 2], [3], [4, 5]], X)).
X = [1, 2, 3, 4, 5].
```