Лекция 9

Символьная алгебра

Преобразование символьного выражения в дерево

..[Op,A,B] - оператор, преобразующий список в структурный терм.
Список содержит имя структурного терма (Op) и аргументы (A,B).

праздник("женский день",8,3)=..[праздник,"женский день",8,3]

?- a+(n+m)=..[Op,A,B].
Op = +
A = a
B = n + m

?- a+n+m=..[Op,A,M].
Op = +
A = a + n
M = m

Ассоциативность операций

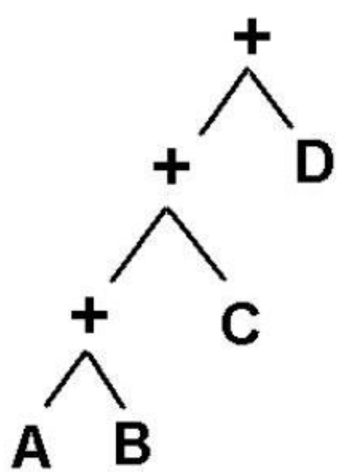
op(Приоритет, Тип, Имя).

Левая ассоциативность:

оператор сложения + определен как
`op( 500, yfx, +).`
+ обладает левой ассоциативностью.
`A + B + C + D`

выполняется как:

`+(+(+(A,B),C),D)`

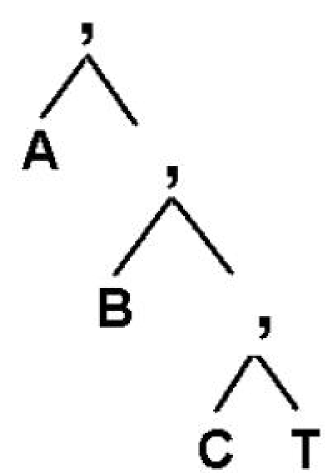


Правая ассоциативность:

Оператор коньюнкции целей определен:
`op( 1100, xfy, ,).`
, обладает правой ассоциативностью.
`A , B , C , D`

выполняется как

`,(A, ,(B, ,(C,D)))`



Дифференцирование многочленов

```
?- d(2*x+3,x,R), simlify(R,R1).
```

```
d(X,X,1):-!
d(C,X,0):-atomic(C).
d(U+V,X,A+B):-d(U,X,A),d(V,X,B).
d(C*X,X,C):-atomic(C), not(C=X).
d(U*V,X,A*V+U*B):-d(U,X,A),d(V,X,B).
```

```
?- d(3*x*x+2*x+1,x,R)
R=3*1*x+3*x*1+2*1+0
```

Упрощение выражений

```
simplify(X,X):-atomic(X).
simplify(X,Y):- X=..[Op,A,B],simplify(A,U),simplify(B,V), rule(Op,U,V,Y).
rule(+,X,0,X).
rule(+,0,X,X).
rule(+,X,Y,Z):-integer(X),integer(Y),Z is X+Y.
rule(+,X,Y,Y+X):-integer(Y). %первым поставить число
rule(+,X,Y,X+Y). %для обратного перевода из списка в оператор

rule(*,X,0,0).
rule(*,X,1,X).
rule(*,0,X,0).
rule(*,1,X,X).
rule(*,X,Y,Z):-integer(X),integer(Y),Z is X*Y.
rule(*,X,Y,Y*X):-integer(Y).%первым поставить число
rule(*,X,Y,X*Y). %для обратного перевода из списка в оператор
```

Использование Пролога в синтаксическом разборе

Порождающей грамматикой называется следующая четверка:

G = <VT, VN, S, P>

где

VT, VN - соответственно, **терминальный** и **нетерминальный** словари,

S - начальный символ,

P = $\{\alpha i \rightarrow \beta i\}$ - множество правил вывода,

причем

α**i** - цепочка, *содержащая* <u>*нетерминальный*</u> символ

β**i** - произвольная цепочка из <u>*терминальных*</u> и <u>*нетерминальных*</u> символов.

Непосредственным порождением называется отношение:

$$\lambda \Rightarrow \mu,$$

где λ = $\delta 1$ $\boldsymbol{\alpha}$**i** $\delta 2$

μ = $\delta 1$ $\boldsymbol{\beta}$**i** $\delta 2$

и существует правило $\boldsymbol{\alpha}$**i** $\rightarrow$ $\boldsymbol{\beta}$**i**.

Порождением называется отношение:

$$\gamma 1 \Rightarrow^* \gamma_n, \qquad n = 1, 2, \ldots,$$

если существует последовательность отношений:

$$\gamma 1 \Rightarrow \gamma 2 \Rightarrow \ldots \Rightarrow \gamma n.$$

Языком, порождаемым грамматикой G, называется:

$$\mathbf{L(G) = \{\gamma \mid S \Rightarrow^* \gamma\},}$$

где γ - терминальная цепочка

(цепочка, состоящая только из элементов терминального словаря VT).

Парсер символьного математического выражения

VT= {0-9, ‘+’ , ‘-‘ , ’*’ , ’:’}

VN={Term, Expr, Number}

S=Expr

P:

- Expr -> Term | Expr +Term | Expr-Term
- Term -> Number | Term*Number | Term :Number
- Number -> 0-9

Грамматика языка (реверсированная)

- Expr -> Term | Term + Expr | Term - Expr
- Term -> Number | Number * Term| Number : Term
- Number -> 0-9

Предикат калькулятора (по строке)

```
?-calculate([5,'+',2,'*',3],E),write(E).

E=11
```

```
calculate (Expr,Val):-reverse(Expr,Expr1),
    a_expr(Expr1,Val).

a_number([NS],NS):-number(NS).

a_term(T,V):- a_number(T,V).

a_term(T,V):-append(X,['*'|Y],T),
            a_number(X,Vx), a_term(Y,Vy), V is Vx*Vy.
```

```
a_term(T,V):-append(X,[':'|Y],T),
                a_number(X,Vx), a_term(Y,Vy),V is Vy/Vx.

a_expr(T,V):-a_term(T,V).

a_expr(T,V):-append(X,['+'|Y],T),
                a_term(X,Vx), a_expr(Y,Vy), V is Vx+Vy.

a_expr(T,V):-append(X,['-'|Y],T),
                a_term(X,Vx), a_expr(Y,Vy), V is Vy-Vx.
```