

Язык SWI Prolog

**Рекурсивное программирование на
языке Пролог**

Использование рекурсии в логическом программировании

Рекурсия в логическом программировании применяется в двух случаях:

- ❖ если отношение описывается с помощью такого же отношения;
- ❖ когда сложный объект (структура) сам является частью однотипного, сложного объекта.

Рекурсивные правила

Отношения на языке Пролог описываются с помощью правил. Правило, содержащее свой заголовок в качестве предиката в правой части этого правила, называется **рекурсивным**. Рекурсивное правило реализует повторяющиеся действия. Рекурсивные правила эффективны при программировании циклических задач, при формировании запросов к базам данных и при обработке списков.

Синтаксис рекурсивных правил и процедур

В общем случае рекурсивная процедура имеет следующий вид:

<заголовок рекурсивного правила>:—<предикат условия выхода>, <предикаты>.

<заголовок рекурсивного правила >:—<предикаты>,

<заголовок рекурсивного правила >,<предикаты>.

<заголовок рекурсивного правила >:—<предикаты>,

<заголовок рекурсивного правила >,<предикаты>.

.....

<заголовок рекурсивного правила >:—<предикаты>,

<заголовок рекурсивного правила >,<предикаты>.

Синтаксис рекурсивных правил и процедур (продолжение)

Для того, чтобы рекурсивная процедура завершалась, необходимо, чтобы в процедуру было включено **условие выхода из рекурсии**, гарантирующее окончание работы процедуры.

Правило, содержащее условие выхода из рекурсии, является нерекурсивным.

Синтаксис рекурсивных правил и процедур (продолжение)

В языке Пролог при согласовании целей правила в процедуре выбираются в порядке их записи в программе. В рекурсивных программах порядок записи правил может оказаться весьма важным. Неудачное расположение правил может привести к бесконечным, рекурсивным вызовам, завершением которых является переполнение стека.

Чтобы обеспечить проверку условий завершения рекурсивных вызовов, рекомендуется нерекурсивное правило с условием выхода записывать перед рекурсивными правилами.

Примеры рекурсивных процедур.

Пример 1. Программа определения суммы
ряда натуральных чисел.

sum_series (1,1).

sum_series(N,S):— $N > 0$, Next is $N-1$,
sum_series(Next,S1), S is $N+S1$.

? — sum_series(6,S).

S=21.

YES

Примеры рекурсивных процедур.

Пример 2. Программа генерации ряда натуральных чисел от N до 20.

write_number (20) :—write(20).

write_number(N):— N<20,write(N),nl,Next is N-1,
write_number (Next).

? — write_number(15).

15

16

17

18

19

20

YES

Схема поиска решений в рекурсивных программах

Для представления формализованной схемы поиска решений будут использоваться следующие обозначения:

- ❖ ТР —текущая резольвента. Первой ТР является исходный вопрос. Каждая следующая резольвента ТР получается путем редукции, т. е. замены самой левой цели в предыдущей резольвенте на тело правила, заголовок которого сопоставим с целью, или путем удаления цели. Если цель сопоставима с фактом, и в том и другом случае вырабатывается подстановка и применяется ко всей ТР.

Схема поиска решений в рекурсивных программах

- ❖ Шаг № — шаг вычисления. Шагом вычислений будем считать действия, выполняемые после определения ТР и заканчивающиеся построением новой ТР или выводом об успехе/неудаче.

Схема поиска решений в рекурсивных программах

- ❖ ТЦ —текущая цель. Текущая цель — это цель, подлежащая согласованию на данном шаге.
- ❖ Пр № — правило, применимое для редукции на определенном шаге.
- ❖ Успех— вывод об успешном вычислении вопроса. Неудача— вывод о неуспешном вычислении вопроса.

Схема поиска решений в рекурсивных программах

- ❖ Откат, Возврат. Отказ — это отказ пользователя от выданного успешного ответа на вопрос, механизм возврата включается принудительно. Возврат — это тупиковая ситуация во время вычисления вопроса, механизм возврата включается автоматически.
- ❖ Так как при рекурсивном обращении к правилу создаются новые экземпляры переменных, будем на каждом шаге добавлять к именам переменных индекс в виде номера шага.

Пример поиска решения в рекурсивной программе

Рассмотрим пример классической рекурсивной процедуры вычисления факториала; эта процедура включает два правила:

`fact (1,1).`

`fact(N,Res):—N>1,N1 is N-1,fact(N1,Res1), Res is
N*Res1.`

Схема вычисления запроса

GOAL: fact(3, Res).

имеет следующий вид:

TP: fact(3, Res).

Шаг 1: ТЦ: fact(3, Res).

Пр1: $3=1 \Rightarrow \text{no}$

Пр2: $3>1$, N11 is 3-1, fact(N11, Res11), Res is Res11*3.

TP: $3>1$, N11 is 3-1, fact(N11, Res11), Res is Res11*3.

Шаг 2: ТЦ: $3>1$.

yes

TP: N11 is 3-1, fact(2, Res11), Res is Res11*3.

Схема вычисления запроса

TP: N11 is 3-1, fact(N11,Res11), Res is Res11*3.

Шаг 3: ТЦ: N11 is 3-1,

Yes

{N11=2}

TP: fact(2,Res11), Res is Res11*3.

Схема вычисления запроса

TP: $\text{fact}(2, \text{Res11})$, Res is $\text{Res11} * 3$.

Шаг 4: TЦ: $\text{fact}(2, \text{Res11})$.

Пр1: $2=1 \Rightarrow \text{no}$

Пр2: $2>1$, N12 is 2-1, $\text{fact}(\text{N12}, \text{Res12})$, Res11 is $\text{Res12} * 2$.

TP: $2>1$, N12 is 2-1, $\text{fact}(\text{N12}, \text{Res12})$, Res11 is $\text{Res12} * 2$,
Res is $\text{Res11} * 3$.

TP: $2>1$, N12 is 2-1, $\text{fact}(\text{N12}, \text{Res12})$, Res11 is $\text{Res12} * 2$,
Res is $\text{Res11} * 3$.

Шаг 5: TЦ: $2>1$.

Yes

TP: N12 is 2-1, $\text{fact}(\text{N12}, \text{Res12})$, Res11 is $\text{Res12} * 2$,
Res is $\text{Res11} * 3$.

Схема вычисления запроса

Шаг 6: ТЦ: N12 is 2-1.

Yes

N12=1

TP: fact(1,Res12), Res11 is Res12*2, Res is Res11*3.

Шаг 7: ТЦ: fact(1,Res12).

Пр1: $1=1 \Rightarrow \text{yes}$

{Res12=1}

TP: Res11 is $1*2$, Res is Res11*3.

Схема вычисления запроса

Шаг 8: ТЦ: Res11 is $1*2$.

Yes

{Res11= 2 }

TP: Res is $2*3$.

Схема вычисления запроса

Шаг 9: ТЦ: Res is $2*3$.

{Res=6}

ТР: (пустая резольвента)

{Res=6}

Успех