Язык SWI Prolog

Обработка списков в программах на языке Пролог. Множества.

Определение понятия множества

Мно́жество — один из ключевых объектов математики, в частности, теории множеств. «Под множеством мы понимаем объединение в одно целое определенных, вполне различимых объектов нашей интуиции или нашей мысли» (Георг Кантор).

Операции над множествами

Основными операциями над множествами являются:

объединение: $A \cup B := \{x \mid x \in A \vee x \in B\}$

пересечение: $A \cap B := \{x \mid x \in A \wedge x \in B\}$

разность: $A \setminus B := \{x \mid x \in A \wedge x \notin B\}$

определение подмножества: $A \subset B := \{x \mid x \in A \Rightarrow x \in B\}$

декартово произведение: $A \times B := \{(a,b) \mid a \in A \wedge b \in B\}$

Представление множеств в в виде списков

Списки —структуры данных, с помощью которых можно представлять множества и графы в программах на языке Пролог.

Множества отличаются от списков тем, что в списках могут быть повторяющиеся элементы, а во множествах —нет.

С другой стороны, в списках порядок следования элементов имеет значения, а множества могут быть неупорядоченными.

Предикат unionset

Предикат unionset определяет операцию объединения двух множеств. Объединением двух множеств X и Y является множество Z, состоящее из элементов, которые принадлежат или множеству X, или множеству Y, или обоим множествам одновременно.

Предикат unionset(X,Y,Z) —истинен, если множество Z является объединением множеств X и Y. Схема отношения этого предиката имеет вид:

unionset(<список>,<список>,<список>).

Декларативное определение предиката unionset

Декларативное описание предиката unionset(X,Y,Z) формулируется следующим образом:

- *Объединение пустого множества с непустым множеством X есть множество X.*
- *Список, определяющий первое множество X можно разделить на голову H и хвост Xs. Если голова первого списка H принадлежит второму списку Y, то рекурсивно вызывается процедура unionset с аргументами Xs и Y. При этом терм H в результирующий список не помещается.*
- *Список, определяющий первое множество X можно разделить на голову H и хвост Xs. Если голова первого списка H не принадлежит второму списку Y, то рекурсивно вызывается процедура unionset с аргументами Xs и Y. При этом терм H помещается в результирующий список.*

Правило unionset

Процедура unuonset(X,Y) состоит из трех правил:

```
unionset([],X,X).
unionset([H|Xs],Y,Z):-member(H,Y),!,unionset(Xs,Y,Z).
unionset([H|Xs],Y,[H|Z]):-unionset(Xs,Y,Z).
```

Предикат interset

Предикат interset определяет операцию пересечения двух множеств. Пересечением двух множеств X и Y является множество Z, состоящее из элементов, которые принадлежат и множеству X и множеству Y одновременно.

Предикат interset (X,Y,Z) —истинен, если множество Z является пересечением множеств X и Y. Схема отношения этого предиката имеет вид:

interset(<список>,<список>,<список>).

Декларативное определение предиката interset

Декларативное описание предиката interset(X,Y,Z) формулируется следующим образом:

Пересечение пустого множества с непустым множеством X есть пустое множество.

Список, определяющий первое множество X, можно разделить на голову H и хвост Xs. Если голова первого списка H принадлежит второму списку Y, то рекурсивно вызывается процедура interset с аргументами Xs и Y. При этом терм H помещается в результирующий список.

Список, определяющий первое множество X, можно разделить на голову H и хвост Xs. Если голова первого списка H не принадлежит второму списку Y, то рекурсивно вызывается процедура interset с аргументами Xs и Y. При этом терм H в результирующий список не помещается.

Процедура interset

```
interset([],X,[]).
interset([H|Xs],Y,
    [H|Z]):-member(H,Y),!,interset(Xs,Y,Z).
interset([H|Xs],Y,Z):-interset(Xs,Y,Z).
```

Предикат difset

Предикат difset определяет операцию разности двух множеств. Разностью двух множеств X и Y является множество Z, состоящее из элементов, которые принадлежат множеству X, но не принадлежат множеству Y.

Предикат difset (X,Y,Z) —истинен, если множество Z является разностью множеств X и Y. Схема отношения этого предиката имеет вид:

difset(<список>,<список>,<список>).

Декларативное определение предиката difset

Декларативное описание предиката difset(X,Y,Z) формулируется следующим образом:

- *Разность пустого множества и непустого множеством X есть пустое множество.*
- *Список, определяющий первое множество X, можно разделить на голову H и хвост Xs. Если голова первого списка H не принадлежит второму списку Y, то рекурсивно вызывается процедура difset с аргументами Xs и Y. При этом терм H помещается в результирующий список.*
- *Список, определяющий первое множество X, можно разделить на голову H и хвост Xs. Если голова первого списка H принадлежит второму списку Y, то рекурсивно вызывается процедура difset с аргументами Xs и Y. При этом терм H в результирующий список не помещается.*

Процедура difset

```
difset([],X,[]).
difset([H|Xs],Y,
    [H|Z]):-not(member(H,Y)),!,difset(Xs,Y,Z).
difset([H|Xs],Y,Z):-difset(Xs,Y,Z).
```

Предикат subset

Предикат subset определяет, является ли множество подмножеством другого множества. Множество X является подмножеством множества Y, если все элементы X принадлежат множеству Y.

Предикат subset(X,Y) —истинен, если множество X является подмножеством Y. Схема отношения этого предиката имеет вид:

subset(<список>,<список>).

Декларативное определение предиката subset

Декларативное описание предиката subset(X,Y) формулируется следующим образом:

Пустое множество является подмножеством любого множества.

Список, определяющий первое множество X, можно разделить на голову H и хвост Xs. Множество X есть подмножество Y, если голова первого списка H принадлежит второму списку Y и Xs есть подмножество множества Y.

Процедура subset

```
subset([],Y).
subset([H|Xs],Y):-member(H,Y),subset(Xs,Y).
```

Предикат dek

Предикат dek определяет операцию декартова произведения двух множеств. Декартовым произведением двух множеств X={xi} и Y={yi} является множество Z, состоящее из пар элементов [xi, yi], где xi принадлежат множеству X, а yi принадлежат множеству Y.

Предикат dek(X,Y,Z) — истинен, если множество Z является декартовым произведением множеств X и Y. Схема отношения того предиката имеет вид:

dek(<список>,<список>,<список списков>).

Предикаты pro и append1

В процедуре dek используются дополнительные предикаты pro и append1.

Предикат pro(X,Y,Z) — истинен, если X есть терм, Y={Yi} — множество термов, а Z является множество пар вида [X, Yi], где X есть заданный терм, а Yi — соответствующий элемент множества Y. Схема отношения предиката pro имеет вид:

pro(<терм>,<список>,<список списков>).

Декларативное определение предиката pro

Декларативное определение предиката pro(X,Y,Z) формулируется следующим образом:

Список Y состоит из одного элемента. Тогда список Z является списком пар вида [X, Yi].

Список, определяющий множество Y, можно разделить на голову H и хвост T. Тогда список Z=[[X,H]|T1], если список T1 есть результат процедуры pro(X,T,T1).

Декларативное определение предиката dek

Декларативное описание предиката dek(X,Y,Z) формулируется следующим образом:

- Список X состоит из одного элемента. Тогда список Z является результатом процедуры pro(X,Y,Z1).
- Список, определяющий первое множество X, можно разделить на голову X и хвост T1. Если pro(X,Y,T2) и dek(T1,Y,T3) и append1(T2,T3,Z) истинны, то Z есть декартово произведение списков X и Y.

Процедура dek

```
append1([],L,L).
append1([H|L],M,[H|R]):- append1(L,M,R).
pro(X,[Y],[[X,Y]|[]]).
pro(X,[H|T], [[X,H]|T1]):-pro(X,T,T1).
dek([X|[]],Y,Z):-pro(X,Y,Z).
dek([X|T1],Y,Z):-pro(X,Y,T2),dek(T1,Y,T3),append1(T2,
    T3,Z).
```

Пример запроса к процедуре dek

Пример запроса к процедуре dek.

? —dek([4,2][3,5,8],Z).

Z=[[4,3],[4,5],[4,8], [2,3],[2,5],[2,8]]

Yes