Язык SWI Prolog

Обработка списков в программах на языке Пролог. Программы сортировки.

Наиболее распространенные методы сортировки списков

- метод сортировки путем прямого выбора;
- метод сортировки путем вставки;
- метод сортировки с использованием перестановок элементов.

Метод прямого выбора

Алгоритм сортировки списка путем прямого выбора включает в себя следующие шаги:

- В исходном списке S1 находится минимальный элемент Min и помещается в результирующий список S2.
- Этот элемент удаляется из списка S1, и процедура поиска повторяется. С каждым шагом список S1 уменьшается.
- Когда список S1 станет пустым, список S2 станет результирующим, упорядоченным по возрастанию списком.

Процедура sort_vibor

Процедура sort_vibor использует следующие процедуры:

- sort1(<исходный список>, <накапливающийся список>, <результирующий список>) — процедура накопления списка;
- delete(<терм>, <список>, <список >) — процедура удаления элемента из списка;
- append(<список>, <список>, <список >) — процедура объединения списков;
- minlist(<список>, <терм>) — процедура определения минимального элемента в списке;
- min(<терм>, <терм>, <терм>) — процедура определения меньшего из двух термов.

Предикат sort_vibor

Предикат sort_vibor(L1,LS)истинен, если список LS получен из списка L1 путем упорядочения списка L1 по возрастанию методом прямого выбора.

Списки L1 и LS могут быть списками числовых термов или символов.

Определения процедуры sort_vibor

```
sort_vibor(L1,LS): —sort1(L1,[],LS).
sort1(L1,L2,LS): —minlist(L1,Min),append(L2,[Min],L3),
      delete(Min,L1,LL),sort1(LL,L3,LS).
sort1([],LS,LS).
delete(A,[A|B],B).
delete(A,[B|T1],[B|T2]):- delete(A,T1,T2).
append([],L,L).
append([X|L1],L2,[X|L3]) : —append(L1,L2,L3).
minlist([X],X).
minlist([X|T],M) : — minlist(T,MinN),min(X,MinN,M).
min(X,Y,X) : —X<=Y.
min(X,Y,Y) : —X>Y.
```

Метод вставки

Алгоритм сортировки списка путем вставки заключается в следующем:

для того, чтобы упорядочить непустой список

L = [X|T], необходимо:

- упорядочить хвост списка, получив список OT;
- вставить голову X списка L в упорядоченный список OT, поместив X в такое место списка OT, что список OT остался бы упорядоченным.

Процедура sort_insert

Процедура sort_insert использует процедуру insert, которая вставляет терм в упорядоченный список, не нарушая упорядочивания.

Предикат insert(X,L1,L2) —истинен, если список L2 получается из упорядоченного списка L1 путем вставки терма X без нарушения порядка. Схема отношения этого предиката имеет вид:

insert(<терм>, <список>, <список>).

Предикат sort_insert(L1,LS) истинен, если список LS получен из списка L1 путем упорядочения списка L1 по возрастанию методом вставки. Списки L1 и LS являются списками числовых термов или символов.

Определения процедуры sort_insert

```
sort_insert([],[]).
sort_insert([X|T],OL): —sort_insert(T,OT),
                        insert(X,OT,OL).
insert(X,[],[X]).
insert(X,[Y|T],[X,Y|T]): —X<=Y.
insert(X,[Y|T],[Y|T1]): —X>Y,insert(X,T,T1).
```

Метод перестановки

Алгоритм сортировки списка путем перестановки элементов заключается в следующем: для того, чтобы упорядочить непустой список L = [X|T], необходимо:

- найти в списке L два смежных элемента X и Y, таких что X>Y,и поменять X и Y местами, получив таким образом новый список L1, затем рекурсивно применить эту процедуру к списку L1;
- если в списке L нет ни одной пары смежных элементов X и Y, таких что X>Y, то список L упорядочен.

Процедура sort_p

Процедура sort_p использует процедуру perest, которая переставляет два смежных элемента в порядке возрастания (или убывания).

Предикат perest(L1,L2) —истинен, если список L2 получается из неупорядоченного списка L1 путем перестановки. Схема отношения этого предиката имеет вид:

perest(<список>, <список>).

Предикат sort_p(L1,LS) истинен, если список LS получен из списка L1 путем упорядочения списка L1 по возрастанию методом перестановки элементов. Списки L1 и LS являются списками числовых термов или символов.

Определения процедуры sort_p

```
sort_p(L,OL):  —perest(L,L1),!,sort_p(L1,OL).
sort_p(OL,OL).
perest([X,Y|T],[Y,X|T]):  —X>Y.
perest([Z|T],[Z|T1]): —perest(T,T1).
```