Programowanie w Logice
Rekurencja i rekurencja ogonowa
(Lista 3)

Przemysław Kobylański

Wstęp

Poniższy predykat `maximum(L, M)` jest spełniony, gdy `M` jest największą wartością na liście liczb `L`:

```
maximum([X], X).
maximum([X | L], M) :-
    maximum(L, M1),
    porównaj(M1, X, M).

porównaj(M, X, X) :-
    X > M.
porównaj(M, X, M) :-
    X =< M.
```

Jest on zapisany rekurencyjnie ale ostatnim wywoływanym w nim warunkiem nie jest `maximum/2` ale `porównaj/3`.

Aby kompilator mógł zoptymalizować przekład i użyć iteracji zamiast rekurencji zużywającej stos, należy przekształcić go do postaci rekurencji ogonowej:

```
maximum([X | L], M) :-
    maximum(L, X, M).

maximum([], M, M).
maximum([X | L], M1, M) :-
    porównaj(M1, X, M2),
    maximum(L, M2, M).

porównaj(M, X, X) :-
    X > M.
porównaj(M, X, M) :-
    X =< M.
```

Predykat `porównaj(M1, X, M2)` można zapisać stosując predykat `->/2`. Służy on do sprawdzania celu, pod warunkiem, że inny cel jest spełniony i ma następującą postać:

```
(IF -> THEN; ELSE)
```

Jeśli spełniony jest warunek IF, to sprawdzany jest warunek THEN, w przeciwnym przypadku sprawdzany jest warunek ELSE.

Predykat `maximum/2` z użyciem predykatu `->/2`:

```
maximum([X | L], M) :-
    maximum(L, X, M).

maximum([], M, M).
maximum([X | L], M1, M) :-
    (X > M1 -> M2 is X; M2 is M1),
    maximum(L, M2, M).
```

Zadania

Zadanie 1 (5 pkt)

Napisz predykat `wariancja(L, D)`, który dla danej listy liczb `L` wylicza wartość wariancji `D`.

Przykład

```
?- wariancja([1, 2, 3], X).
X = 0.666666666666667.

?- wariancja([1, 2, 3, 4, 5], X).
X = 2.

?- wariancja([1, 2, 3, 4, 5, 6, 7, 8], X).
X = 5.25.
```

Zadanie 2 (3 pkt)

Dana jest lista liczb całkowitych $L = [a_0, \ldots, a_{n-1}]$. Przez sekcję $a[i : j]$, gdzie $0 \leq i \leq j < n$, rozumiemy fragment listy złożony z elementów $a_i, a_{i+1}, \ldots, a_j$. Suma sekcji $a[i : j]$ jest równa $\sum_{k=i}^{j} a_k$. Przyjmij, że sumą pustej sekcji jest wartość 0.

Napisz predykat `max_sum(L, S)`, który dla danej listy L znajduje największą wartość S spośród wszystkich sum po wszystkich możliwych sekcjach.

Uwaga

Aby zaliczyć to zadanie, predykat `max_sum` powinien być zapisany w postaci rekurencji ogonowej oraz jego czasowa złożoność obliczeniowa powinna być liniowa w stosunku do długości danej listy.

Przykład

```
?- max_sum([3, -5, 4, -1, 3, -5, 2], X).
X = 6.
```

Zadanie 3 (2 pkt)

Napisz program dla `even_permutation(Xs, Ys)` i `odd_permutation(Xs, Ys)`, który znajduje listę `Ys` będącą, odpowiednio, parzystą i nieparzystą permutacją listy `Xs`.

Ćwiczenie 3.3.1(iv) z [2].

Przykład

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

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

Literatura

[1] W.F. Clocksin, C.S. Mellish. Prolog. Programowanie. Helion, 2003.

[2] L. Sterling, E. Shapiro. The Art of Prolog. The MIT Press, 1994.