Universidad de Huelva

PROBLEMA NO. 93
PUZLE ARITMÉTICO

VICTOR M. RODRÍGUEZ NAVARRO
REPRESENTACIÓN DEL CONOCIMIENTO
ETSI Universidad de Huelva

Problema no. 93: Puzle Aritmético – Víctor M. Rodríguez Navarro

Este documento detalla la resolución del problema número 93 de la plataforma P-99 (Ninety-nine ProLog Problems). Idea original del problema por Roland Beuret.

Planteamiento

Dada una lista de números enteros, encontrar la forma correcta de insertar los signos aritméticos de tal forma que el resultado sea una ecuación válida, es decir, que se cumpla la igualdad (ambos términos sean equivalentes).

Ejemplo

Con la lista de números [1,2,3,4] podríamos formar las siguientes ecuaciones:

- 1 = 2+(3-4)
- 1 = 2+3-4
- 1-2 = 3-4

Implementación

A continuación, se detallan los predicados implementados para el funcionamiento del algoritmo.

equation(+L, -LT, -RT)

Es cierto si LT y RT forman la lista de números enteros L y además ambos términos de la ecuación (izquierdo y derecho) son equivalentes.

Este predicado se usa en la llamada principal, ya que construye los términos izquierdo y derecho de la ecuación y los evalúa.

```
?- equation([1,2,3,4], LT, RT).
LT = 1,
RT = 2+(3-4) ;
LT = 1,
RT = 2+3-4 ;
LT = 1-2,
RT = 3-4 ;
false.
```

term(+L, -T)

Es cierto si T es un término formado a partir de la lista de números enteros L.

Este predicado se usa para construir todos los términos posibles a partir de la lista L.

```
?- term([1,2,3], T).
T = 1+(2+3) ;
T = 1-(2+3) ;
T = 1*(2+3) ;
T = 1/(2+3) ;
T = 1+(2-3) ;
T = 1-(2-3) ;
T = 1*(2-3) ;
...
```

binterm (+LT, +RT, -T)

Es cierto si T es el término binario construido a partir de intercalar un operador aritmético entre los términos LT y RT.

Este predicado se usa para construir términos binarios en el predicado *term*.

```
?- binterm(1, 2, T).
T = 1+2 ;
T = 1-2 ;
T = 1*2 ;
T = 1/2.
```

split(-L, +L1, +L2)

Es cierto si L1 y L2 no están vacíos y forman la lista L.

Este predicado se usa para dividir en todas las partes posibles la lista L en el predicado *equation.*

Nótese el uso reversible de append

```
?- split([1,2,3,4], L1, L2).
L1 = [1],
L2 = [2, 3, 4] ;
L1 = [1, 2],
L2 = [3, 4] ;
L1 = [1, 2, 3],
L2 = [4] ;
```

do(+L)

Es cierto si se imprimen por pantalla todas las soluciones del problema, dada una lista de números L.

Este predicado se usa como llamada principal para resolver el problema, ya que muestra las soluciones obtenidas en el predicado *equation* línea a línea. A continuación, podemos ver la resolución del problema con la lista de entrada L = [5, 3, 3, 5].

```
?- do([5, 3, 3, 5]).
5 = 3-(3-5)
5 = 3/(3/5)
5 = 3-3+5
5 = 3/3*5
5+3 = 3+5
5*3 = 3*5
5+(3-3) = 5
5-(3-3) = 5
5*(3/3) = 5
5/(3/3) = 5
5+3-3 = 5
5-3+3 = 5
5*3/3 = 5
5/3*3 = 5
true.
```