4. Use a lei da troca para exprimir o isomorfismo undistl $= [i_1 \times id, i_2 \times id]$ sob a forma de um 'split' de alternativas.

$$(B + C) \times A \xleftarrow{\text{undistl}} (B \times A) + (C \times A)$$

Resolução

$[i_1 \times id, i_2 \times id]$

{ def-x, **lei (10)** }

$[< i_1 \cdot \pi_1, id \cdot \pi_2 >, < i_2 \cdot \pi_1, id \cdot \pi_2 >]$

{ lei da troca, **lei (28)** }

$< [i_1 \cdot \pi_1, i_2 \cdot \pi_1], [id \cdot \pi_2, id \cdot \pi_2] >$

{ natural-id, **lei (1)** }

$< [i_1 \cdot \pi_1, i_2 \cdot \pi_1], [\pi_2, \pi_2] >$

{ def-+, **lei (21)** }

$< \pi_1 + \pi_1, [\pi_2, \pi_2] >$

Haskell

In [1]:
```
:load ../src/Cp.hs

-- type checking

:t split (p1 -|- p1) (either p2 p2)
```

```
split (p1 -|- p1) (either p2 p2) :: forall a1 c a2. Either (a1, c)
(a2, c) -> (Either a1 a2, c)
```