Cálculo de Programas

2.° ano
Lic. Ciências da Computação e Mestrado Integrado em Engenharia Informática
UNIVERSIDADE DO MINHO

2020/21 - Ficha nr.° 3

1. Considere o isomorfismo

$$(A+B)+C \underset{\text{coassocl}}{\overset{\text{coassocr}}{\cong}} A+(B+C)$$

onde coassocr $= [id+i_1, i_2 \cdot i_2]$. Calcule a sua conversa resolvendo em ordem a coassocl a equação,

$$\text{coassocl} \cdot \text{coassocr} = id$$

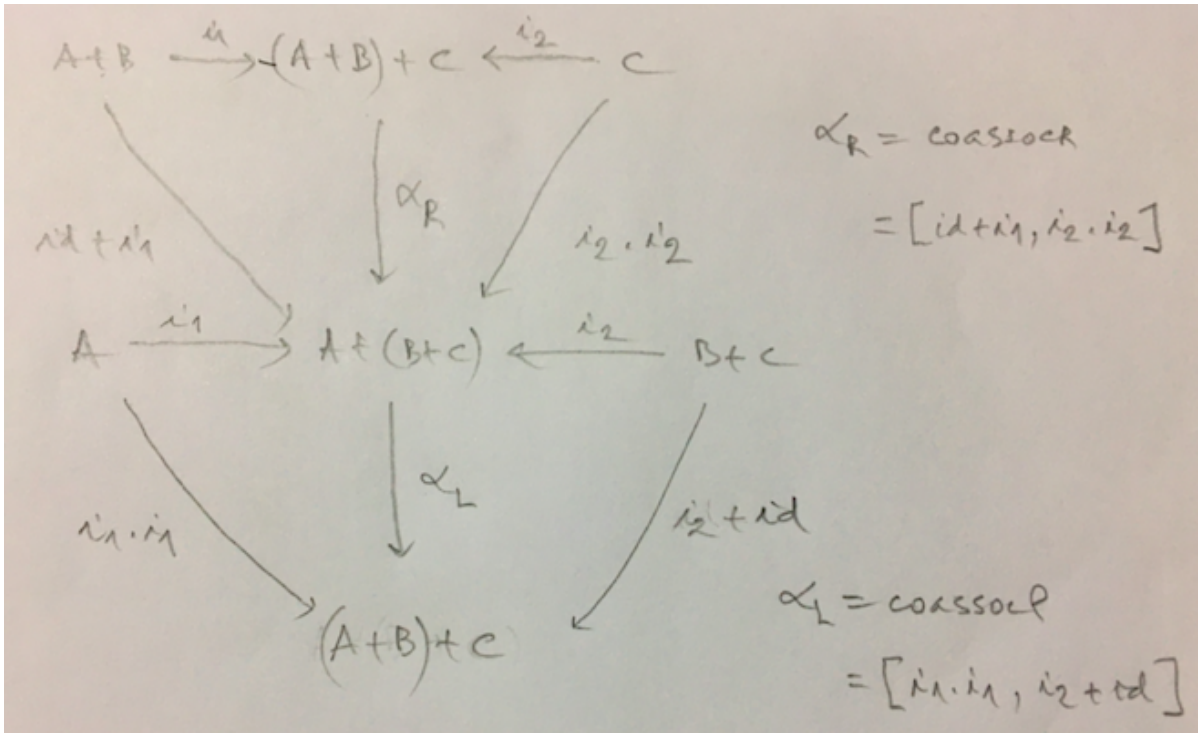
isto é

$$\text{coassocl} \cdot \underbrace{[id+i_1, i_2 \cdot i_2]}_{\text{coassocr}} = id$$

etc. Finalmente, exprima coassocl sob a forma de um programa em Haskell *não recorra* ao combinador "either".

Resolução

Vamos começar por desenhar o diagrama que testemunha o isomorfismo.



Sendo $coassocl = [i_1 \cdot i_1, i_2 + id]$, vamos então demonstrar que $coassocl \cdot coassocr = id$

$[i_1 . i_1, i_2 + id] . [id + i_1, i_2 . i_2]$

{ fusão-+, **lei (20)** }

$$[[i_1 \cdot i_1, i_2 + id] \cdot (id + i_1)\ ,\ [i_1 . i_1, i_2 + id] \cdot (i_2 \cdot i_2)]$$

{ assoc-comp, **lei (2)** }

$$[[i_1 \cdot i_1, i_2 + id] \cdot (id + i_1)\ ,\ ([i_1 . i_1, i_2 + id] \cdot i_2) . i_2)]$$

{ cancelamento-+, **lei (18)** }

$$[[i_1 \cdot i_1, i_2 + id] \cdot (id + i_1)\ ,\ (i_2 + id) \cdot i_2)]$$

{ natural-i_2, **lei (24)** }

$$[[i_1 \cdot i_1, i_2 + id] \cdot (id + i_1)\ ,\ i_2 \cdot id]$$

{ absorção-+, **lei (22)**; natural-id, **lei(1)** }

$$[[i_1 \cdot i_1 \cdot id, (i_2 + id) \cdot i_1]\ ,\ i_2]$$

{ natural-id, **lei(1)** ; natural_i_1, **lei (23)** }

$$[[i_1 \cdot i_1, i_1 \cdot i_2]\ ,\ i_2]$$

{ fusão-+, **lei (20)** }

$$[i_1 \cdot [i_1, i_2]\ ,\ i_2]$$

{ reflexão-+, **lei (19)** }

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

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

$$[i_1\ ,\ i_2]$$

{ reflexão-+, **lei (19)** }

$$id$$

Haskell

In [1]:

```
:load ../src/Cp.hs

-- pointfree

coassoclPF = either (i1 . i1) (i2 -|- id)

-- pointwise

coassoclPW (Left a) = Left . Left $ a
coassoclPW (Right (Left a)) = Left . Right $ a
coassoclPW (Right (Right a)) = Right a
```

In [2]:

```
-- type checking

:t coassoclPF
:t coassoclPW
```

```
coassoclPF :: forall a b1 b2. Either a (Either b1 b2) -> Either
(Either a b1) b2
coassoclPW :: forall a b1 b2. Either a (Either b1 b2) -> Either
(Either a b1) b2
```