Raciocínio

O que queremos provar:

- Para toda a lista finita não vazia de floats xs, o máximo da lista xs é igual ao máximo dessa mesma lista pela ordem inversa.

Isto é:

```
maximum' xs == maximum' (reverse’ xs)
```

Usando a função maximum’, reverse’, max’ , que são definidas recursivamente no código que se segue, e a lei dada no enunciado:

```
maximum' :: Ord float => [float] -> float
maximum' (x:xs) = maximum'' xs x -- maximum' Eq 1
-> tendo em conta que não é uma lista não vazia

maximum'' :: Ord float => [float] -> float -> float
maximum'' [] y = y -- maximum'' Eq 1
maximum'' (x:xs) y = maximum'' xs (max' x y)-- maximum'' Eq 2

max' :: Ord float => float -> float -> float
max' x y
    |x > y = x -- max Eq 1
    |otherwise = y -- max Eq 2

reverse' :: [a] -> [a]
reverse' [] = [] -- reverse Eq 1
reverse' (x:xs) = reverse' xs ++ [x] -- reverse Eq 2

maximum (ps ++ qs) = maximum ps `max` maximum qs -- given law
```

Caso base ([x]):

```
maximum’ [x] = -- abbreviation
maximum’ x : [ ] = -- maximum' Eq 1
maximum’’ [ ] x = -- maximum'' Eq 1
x

maximum’ (reverse [x]) = -- reverse' Eq 2
maximum' (reverse [] ++ [x]) = -- reverse' Eq 1
maximum’ [x] = -- abbreviation
maximum’ x : [ ] = -- maximum' Eq 1
maximum [ ] x = -- maximum'' Eq 1
x
```

x == x, como queríamos demonstrar

Caso de indução (x:xs) :

```
maximum' (x:xs) = -- abbreviation
maximum' [x] ++ xs = -- maximum' Eq 1
maximum [x] `max` maximum xs = -- given law
x `max` maximum xs

maximum ' (reverse' (x:xs)) = -- maximum' Eq 1
maximum' (reverse' xs ++ [x]) = -- reverse' Eq 2
maximum' reverse' xs `max` maximum [x] = -- given law
maximum' reverse' xs `max` x = -- IH
x `max` maximum' xs
```

x \`max\` maximum’ xs == x \`max\` maximum’ xs, como queríamos demonstrar

Concluindo, por indução, está provado que maximum’ xs == maximum’ (reverse’ xs)

André Mendes fc54453
Filipa Almendra fc54396
PP-2020/2021