Higher Order Operator Overloading and Extensional Equality

by Sven Nilsen, 2020

When two functions return the same output for every input,
they are said to be “extensionally equal” or “logically equivalent”:

f <=> g   <=>   ∀ x { f(x) == g(x) }

The problem is that checking this for every input can be very expensive.
In some cases one is able to use algebra to prove this more efficiently.
What is the minimum requirement of existential knowledge to prove this with algebra?

Under Higher Order Operator Overloading (HOOO), one can use the following law:

∴   ((f == g) <=> \true) == (f <=> g)

∵   f : T → U
∵   g : T → U
∵   (f == g) : T → bool

If the function `f == g` is extensionally equal to `\true`, then `f` and `g` are extensionally equal.

Therefore, the minimum requirement of existential knowledge is:

\true <=> \true

From this, together with HOOO and algebra, all other extensional equalities can be inferred.

For example, `dec · inc <=> id` can be proven:

dec · inc == id
\(x : real) = dec(inc(x))   ==   \(x : real) = x
\(x : real) = dec(inc(x)) == x
\(x : real) = dec(x + 1) == x
\(x : real) = (x + 1) – 1 == x
\(x : real) = x + (1 – 1) == x
\(x : real) = x + 0 == x
\(x : real) = x == x
\(x : real) = true
\true

inc(x : real) = x + 1
dec(x : real) = x – 1
id(x : real) = x