

1 Equality of functions

Recall from class that functions are equal if and only if they are equal on all inputs (this equality is called extensionality.)

Definition 1.1. (extensionality) If $f, g :: a \rightarrow b$ then they are defined to be (extensionally) equal as follows:

$$f = g \stackrel{\text{def}}{=} \forall x : a. f(x) = g(x)$$

So, we can prove two functions f and g are equal by choosing an arbitrary x of type a and showing $f(x) = g(x)$.

For example, if $f(x) = |x|$ (the absolute value) and $g(x) = x$ then, $f \neq g$ when we consider them as functions in the type $\mathbb{Z} \rightarrow \mathbb{Z}$ since $f(-2) = 2$ and $g(-2) = -2$. But, if we think of these functions as elements of $\mathbb{N} \rightarrow \mathbb{N}$, they are equal. To see this, choose an arbitrary $x \in \mathbb{N}$ and argue that $f(x) = g(x)$ i.e. that $|x| = x$. But this is trivially true when $x \geq 0$, which follows because $x \in \mathbb{N}$.

Recall the following Haskell definitions.

Definition 1.2. plus

$$\begin{aligned} \text{plus} &:: (\text{Integer}, \text{Integer}) \rightarrow \text{Integer} \\ \text{plus}(x, y) &= x + y \end{aligned}$$

Definition 1.3. plusc

$$\begin{aligned} \text{plusc} &:: \text{Integer} \rightarrow (\text{Integer} \rightarrow \text{Integer}) \\ \text{plusc } x \ y &= x + y \end{aligned}$$

Definition 1.4. curry

$$\begin{aligned} \text{curry} &:: ((a, b) \rightarrow c) \rightarrow (a \rightarrow (b \rightarrow c)) \\ \text{curry } f \ x \ y &= f \ (x, y) \end{aligned}$$

Definition 1.5. uncurry

$$\begin{aligned} \text{uncurry} &:: (a \rightarrow (b \rightarrow c)) \rightarrow ((a, b) \rightarrow c) \\ \text{uncurry } f \ (x, y) &= f \ x \ y \end{aligned}$$

In class we proved the following theorem:

Theorem 1.1.

$$\text{curry plus} = \text{plusc}$$

Proof: Note that both *curry plus* and *plusc* have the type $Integer \rightarrow (Integer \rightarrow Integer)$ i.e. they are functions mapping an *Integers* to a function of type $Integer \rightarrow Integer$. This means we can use extensionality to prove they are equal as functions. We must show the following.

$$\forall x : Integer. \text{curry plus } x = \text{plusc } x$$

Assume x is an arbitrary *Integer*. Then we must show

$$\text{curry plus } x = \text{plusc } x$$

But *curry plus x* and *plusc x* are functions of type $Integer \rightarrow Integer$. To show they are equal we use extensionality a second time, to show:

$$\forall y : Integer. \text{curry plus } x \ y = \text{plusc } x \ y$$

We choose an arbitrary y of type *Integer* and show the following

$$\text{curry plus } x \ y = \text{plusc } x \ y$$

Starting with the left side of the equality we get the following:

$$\text{curry plus } x \ y \stackrel{\langle\langle \text{def. of curry} \rangle\rangle}{=} \text{plus } (x, y) \stackrel{\langle\langle \text{def. of plus} \rangle\rangle}{=} x + y$$

On the right side of the equality, we have the following:

$$\text{plusc } x \ y \stackrel{\text{def. of plusc}}{=} x + y$$

Since both sides of the equality are equal to $x + y$ we see that the functions are equal.

□

Problem 1.1. Prove the following theorem using extensionality.

Theorem 1.2. [uncurry-plusc]

$$\text{uncurry plusc} = \text{plus}$$

Hint: The functions (*uncurry plusc*) and *plus* have the type $(Integer, Integer) \rightarrow Integer$. Extensionality for functions f and g of this type can most conveniently be written as

$$\forall (x, y) : (Integer, Integer). f(x, y) = g(x, y)$$

2 Function Composition

Now, consider the following two definitions.

Definition 2.1. Function Composition

$$\begin{aligned} \text{compose} &:: (b \rightarrow c) \rightarrow (a \rightarrow b) \rightarrow (a \rightarrow c) \\ \text{compose } f \ g \ x &= f(g \ x) \end{aligned}$$

In Haskell, $(\text{compose } f \ g)$ is written $(f \cdot g)$, we will write $(f \circ g)$ here.

Definition 2.2. Identity function

$$\begin{aligned} \text{id} &:: a \rightarrow a \\ \text{id } x &= x \end{aligned}$$

Theorem 2.1. Compose-id-right

$$\forall f : a \rightarrow b. (f \circ \text{id}) = f$$

Proof: Choose an arbitrary function $f :: a \rightarrow b$ and show that

$$f \circ \text{id} = f$$

Since f has type $a \rightarrow b$ and id has type $a \rightarrow a$ we can see that $f \circ \text{id}$ has the same type. (why?) We use extensionality to show that these two functions are equal, *i.e.* we must show:

$$\forall x : a. (f \circ \text{id}) x = f x$$

Chose an arbitrary x in type a and show

$$(f \circ \text{id}) x = f x$$

By definition of compose and definition of id we get the following sequence of equalities.

$$(f \circ \text{id})x = \text{compose } f \ \text{id } x = f(\text{id } x) = f x$$

This completes the proof.

□

Problem 2.1. Prove the following theorem.

Theorem 2.2. [compose-id-left]

$$\forall f : a \rightarrow b. \text{id} \circ f = f$$

Hint: First argue that $\text{id} \circ f$ and f have the same type (note that $\text{id} :: b \rightarrow b$) and then use extensionality.