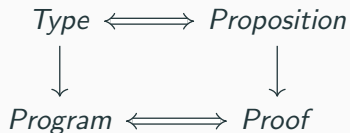


Dependently Typed Programming

The Curry-Howard Correspondence



Philip Wadler. Propositions As Types.

Commun. ACM, 58(12):75–84, November 2015

Types are (usually):

- `Int`
- `String`
- ...

How are these propositions?

Existential Proofs

So when you see:

$$x : \mathbb{N}$$

Think:

$$\exists . \mathbb{N}$$

Remark

We'll see a more powerful and precise version of \exists later.

Proof is “by example”

$$x = 1$$

Example “Proof”

Let's start working with a function as if it were a proof.

The example function we'll choose gets the first element of a list and returns it (commonly called **head** in functional programming languages).

Here's the type:

$$\{A : \text{Set}\} \rightarrow \text{List } A \rightarrow A$$

A Polynomial Solver

The p -Adics
