Type Theory Set - 1

Jason Schuchardt

July 4, 2019

Type theory is built on many years of work from areas of math and computer science that aren't usually studied in high school or the typical undergraduate math curriculum. In particular, the simply typed lambda calculus is built on the notation and ideas of the (untyped) lambda calculus. (`https://en.wikipedia.org/wiki/Lambda_calculus`)

Definition 0.1. We should begin by defining the untyped lambda calculus. The untyped lambda calculus is essentially a syntactic game, i.e., we form expressions according to certain rules, and manipulate (evaluate) them according to other rules.

Formation rules. A lambda expression is recursively defined, and consists of one of the following things:

1. A variable, denoted by a lowercase latin letter, with an optional subscript, e.g.

$$a, x_1, x_2, v, y, w$$

2. A function definition: if x is any variable, and L is a lambda expression, then

$$\lambda x.L$$

   is a lambda expression, which we think of as a function which takes a variable x, and is defined by the expression L. This is also called λ abstraction.

3. Function application. If L and M are lambda expressions, then

$$(L\ M)$$

   is a lambda expression that we think of as representing the application of the function defined by the expression L to the expression M.

Evaluation rules. Lambda expressions may be evaluated

(1)