Types and Lambda Calculus Notes

by Tyler Wright

github.com/Fluxanoia fluxanoia.co.uk

These notes are not necessarily correct, consistent, representative of the course as it stands today or, rigorous. Any result of the above is not the author's fault.

Contents

1 The Axioms of Lambda Calculus

We suppose that we have a countably infinite set of variables $\mathbb{V}$ (we usually refer to elements of this set as x, y, z, etc.), from this we define the alphabet of lambda calculus $\mathbb{V} + \{\lambda, ., (,)\}$.

The set of terms of lambda calculus Λ is defined inductively by the axioms. A string is a term if and only if there is a proof tree using the following axioms that concludes it is a term.

1.1 Axiom of Variables

For some x in $\mathbb{V}$, we have the axiom:

$$\frac{}{x \in \Lambda},$$

which says that each variable is a term.

1.2 Axiom of Application

We have the axiom:

$$\frac{M \in \Lambda \qquad N \in \Lambda}{(MN) \in \Lambda},$$

which says that for some terms M and N, (MN) is also a term.

1.3 Axiom of Abstraction

For some x in $\mathbb{V}$, we have the axiom:

$$\frac{M \in \Lambda}{(\lambda x.M) \in \Lambda},$$

which says that for some term M, $(\lambda x.M)$ is also a term.

1.4 Subterms

Subterms of a term M are substrings of M that are themselves terms and not captured by a λ (directly preceeded by).

1.5 Syntactical Conventions

Parentheses allow our lambda calculus to be unambigious, but for the sake of simplicity, we will construct conventions that will allow us to retain unique meaning with less parentheses:

- Omit outermost parentheses,
- Terms associate to the left, (MNP) parses as $((MN)P)$,
- Bodies of abstractions end at parentheses, $(\lambda x.MN)$ parses as $(\lambda x.(MN))$,
- Group repeated abstractions, $(\lambda xy.M)$ parses as $(\lambda x.(\lambda y.M))$.