Lecture Notes for Functional Programming (COSC 3015)

James Caldwell

September 15, 2015

1 Lambda Terms

A lambda term has the form $(\backslash y \to t[y])$ where $t[y]$ is an arbitrary term, possibly containing free occurrences of the variable y. This free occurrence business is a bit tricky, perhaps best first explained by a few examples. But first, we need to describe how to make the computational step of applying a function (presented as a lambda term) to an argument (which can be any other lambda term), say $\hat{t}$.

$$(\backslash y \to t[y])\,\hat{t} \implies t[\,\hat{t}\,]$$

In this notation, $t[\,\hat{t}\,]$ is the term that results from substituting $\hat{t}$ for all the free occurrences of y in t. Thus:

$$\begin{array}{ll} i.) & (\backslash y \to y)\,5 \Longrightarrow 5 \\ ii.) & (\backslash y \to y)\,\text{"}xyzzy\text{"} \Longrightarrow \text{"}xyzzy\text{"} \\ iii.) & (\backslash y \to \backslash x \to y + x)\,5 \Longrightarrow (\backslash x \to 5 + x)) \\ iv.) & (\backslash y \to \backslash y \to y)\,5 \Longrightarrow (\backslash y \to y) \end{array}$$

Example $(i.)$ is easy, in this case, the term $t[y]$ is just the variable y and so just becomes 5. Example $(ii.)$ is the same but the argument is a string "$xyzzy$". In example $(iii.)$, $t[y]$ is the term $(\backslash x \to y + x)$ and y is free in this term, because there is no lambda binding of y. Thus, the free occurrence of y is replaced by the argument 5 yielding the term $(\backslash x \to 5 + x)$. Example $(iv.)$ is a bit wierd. In the term $(\backslash y \to \backslash y \to y)$, $t[y]$ is the term $(\backslash y \to y)$

The leftmost y in this term is the binding occurence of y. The second y in the term is "bound" by the lambda expression and so we say:

There are no free occurences of y *in the term* $(\backslash y \to y)$.