

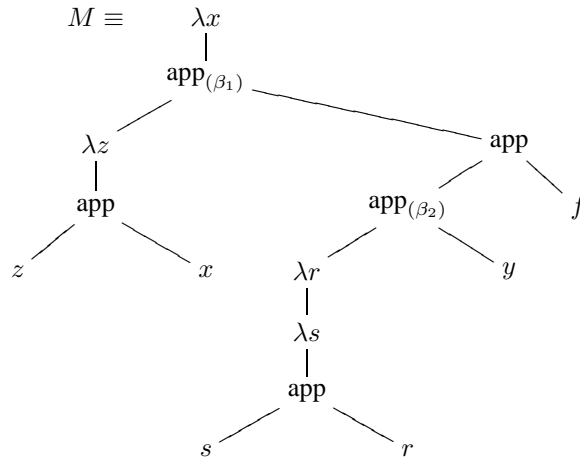
## Solutions for Exercise Sheet 3

### Exercise 1: beta redexes

Given below is a term  $M$ :

$$M = \lambda x. ((\lambda z. z x)((\lambda r. \lambda s. s r) y f))$$

a. Draw the syntax tree for  $M$ .



b. Identify two separate beta redexes in the term  $M$ . The subtrees at  $\beta_1$  and  $\beta_2$  in the tree above.

c. Show the results of reducing  $M$  at each of these redexes. Call the results of these reductions  $N_1$  and  $N_2$ .

$$\begin{aligned} \text{Reduce } M \text{ at } \beta_1: & \quad \lambda x. ((\lambda z. z x)((\lambda r. \lambda s. s r) y f)) \longrightarrow_{\beta} \lambda x. (\lambda r. \lambda s. sr) y f x \quad (\stackrel{\text{def}}{=} N_1) \\ \text{Reduce } M \text{ at } \beta_2: & \quad \lambda x. ((\lambda z. z x)((\lambda r. \lambda s. s r) y f)) \longrightarrow_{\beta} \lambda x. (\lambda z. z x)((\lambda s. sy) f) \quad (\stackrel{\text{def}}{=} N_2) \end{aligned} \quad (\text{For clarity you might prefer to write } N_1 \text{ as } \lambda x. ((\lambda r. \lambda s. sr) y f) x, \text{ but the syntax tree is the same.})$$

In this solution set, we always underline the redex that is being reduced at each stage. This is a good idea for you to do well, in order to be clear about the reductions you are performing.

d. Reduce  $N_1$  and  $N_2$  to normal forms and check if they reduce to the same normal form.

For $N_1$ :	For $N_2$ :
$\lambda x. (\lambda r. \lambda s. sr) y f x$	$\lambda x. (\lambda z. z x)((\lambda s. sy) f)$
$\longrightarrow_{\beta} \lambda x. (\lambda s. sy) f x$	$\longrightarrow_{\beta} \lambda x. (\lambda z. z x)(f y)$
$\longrightarrow_{\beta} \lambda x. f y x$	$\longrightarrow_{\beta} \lambda x. f y x$

e. Specify if your redexes in step b are outermost or innermost redexes.

$\beta_1$  is an outermost redex and not an innermost.  $\beta_2$  is an innermost redex and not an outermost.

f. What are the pro's and con's of always choosing outermost redexes for beta reduction? Similarly, for innermost redexes?

Pro outermost: Reducing outermost redexes can save work (sometimes an infinite amount of work) because it avoids reducing the arguments of  $\lambda$  functions. If a  $\lambda$  function is constant in its argument, then there is no need to reduce the argument because it will get discarded.

Con outermost: If a  $\lambda$  function has multiple occurrences of its argument, then the argument will need to be reduced multiple times. This can add work.

Pro innermost: Every argument term is evaluated exactly once. This saves work if functions use their arguments multiple times, e.g.,  $\lambda y. f y y$ .

Con innermost: This same feature does unnecessary work if the function doesn't use its argument at all.

## Exercise 2: beta reduction

Carry out 2–3 steps of beta-reduction for the following term:

$$(\lambda x. x x x) (\lambda x. x x x)$$

and describe the computational effect of this term.

$$\begin{aligned} & (\lambda x. xxx)\lambda x. xxx \\ \longrightarrow_{\beta} & \frac{(\lambda x. xxx)(\lambda x. xxx)\lambda x. xxx}{(\lambda x. xxx)(\lambda x. xxx)(\lambda x. xxx)\lambda x. xxx} \end{aligned}$$

The computation “diverges” (fails to terminate), the size of the term growing linearly in the number of  $\beta$ -reductions.

## Exercise 3: Church encodings

Use outermost reduction, i.e., reduction by always choosing the outermost redexes, to evaluate the following terms:

a. and true false

$$\begin{aligned} & \longrightarrow_{\beta}^* \text{if true false false} \\ & \longrightarrow_{\beta}^* \text{true false false} \\ & \longrightarrow_{\beta}^* \text{false} \end{aligned}$$

(Note that each of these steps involves 2-3 beta reductions because the function terms have multiple  $\lambda$ 's. We have used  $\longrightarrow_{\beta}^*$  to signify this fact.)

b. and false true

$$\begin{aligned} & \longrightarrow_{\beta}^* \text{if false true false} \\ & \longrightarrow_{\beta}^* \text{false false false} \\ & \longrightarrow_{\beta}^* \text{false} \end{aligned}$$

c. and true (and true false)

$$\begin{aligned} & \longrightarrow_{\beta}^* \text{if true (and true false) false} \\ & \longrightarrow_{\beta}^* \text{true (and true false) false} \\ & \longrightarrow_{\beta}^* \text{and true false} \\ & \longrightarrow_{\beta}^* \text{false} \end{aligned} \quad (\text{by (a)})$$

d. and false (and true false)

$$\begin{aligned} & \longrightarrow_{\beta}^* \text{if false (and true false) false} \\ & \longrightarrow_{\beta}^* \text{false (and true false) false} \\ & \longrightarrow_{\beta}^* \text{false} \end{aligned}$$

Finally, prove or disprove the equality  $\text{and } x y = \text{and } y x$ .

To say that the equality is true is to say that  $\text{and } M N \equiv \text{and } N M$  for all terms  $M$  and  $N$ . This is *false*.

Even though the equivalence holds if **true** or **false** are used as  $M$  and  $N$  (which is a straightforward verification), it fails if other terms are used for  $M$  and  $N$ . The most important case is if  $M$  is  $\Omega = (\lambda x. xx) (\lambda x. xx)$ , the nonterminating term. In this case,

$$\begin{aligned} & \text{and } \Omega \text{ false} \\ & \longrightarrow_{\beta}^* \text{if } \Omega \text{ false false} \\ & \longrightarrow_{\beta}^* \Omega \text{ false false} \end{aligned}$$

Since  $\Omega$  reduces to itself, no further progress is possible. However,  $(\text{and } \text{false } \Omega) \longrightarrow_{\beta}^* \text{false}$ . These two terms are not equivalent. This problem arises because “and” uses “short circuit” evaluation.