The University of Birmingham
School of Computer Science

Exercise Sheet 3

Attempt as many questions as you can during the exercise class, and work on the remainder at home. Hand in your solutions to the tutor in the Friday exercise class, on 12 Feb, 2010.

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.

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

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

e. A redex is said to be *outermost* if it is not a subterm of any other redex. Similarly, it is said to be *innermost* if it does not a have a redex as a subterm. Specify if your redexes in step b are outermost or innermost redexes.

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

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.

Exercise 3: Y combinator

For this exercise, we use an applied lambda calculus with booleans, integers and cons-pairs as primitives. The δ-conversion rules for the cons-pairs are the following:

$$\begin{array}{rcl} \text{null nil} & = & \text{true} \\ \text{null (cons } x\ y) & = & \text{false} \\ \text{car (cons } x\ y) & = & x \\ \text{cdr (cons } x\ y) & = & y \end{array}$$

We feel free to use the normal infix notation for arithmetic operators, e.g, $n + (m * 2)$, instead of the official notation $+\, n\, (*\, m\, 2)$.
Recall the definition of the **Y** combinator:

$$\mathbf{Y}\, t = (\lambda x.\, t\ (x\ x))\ (\lambda x.\, t\ (x\ x))$$

and the fact that beta-reduction gives $\mathbf{Y}\, t \longrightarrow_\beta t\ (\mathbf{Y}\, t)$. We have dubbed this reduction the "DNA" rule.

a. Consider the function:

$$N = \lambda f.\, \lambda n.\, \text{cons}\ n\ (f\ (n+1))$$

Carry out a few beta reductions for the term $\mathbf{Y}\ N\ 1$ and describe its computational effect.

b. Consider the following functions:

$$\begin{array}{rcl} A & = & \lambda f.\, \lambda y.\, \lambda z.\, \text{cons}\ ((\text{car}\ y) + (\text{car}\ z))\ (f\ (\text{cdr}\ y)\ (\text{cdr}\ z)) \\ F & = & \lambda l.\, \text{cons}\ 1\ (\text{cons}\ 1\ (\mathbf{Y}\ A\ l\ (\text{cdr}\ l))) \end{array}$$

Here are the first couple of reduction steps for the term $\mathbf{Y}\ F$.

$$\begin{array}{rcl} \mathbf{Y}\, F & \longrightarrow & F\ (\mathbf{Y}\, F) \\ & \longrightarrow & \text{cons}\ 1\ (\text{cons}\ 1\ (\mathbf{Y}\ A\ (\mathbf{Y}\ F)\ (\text{cdr}\ (\mathbf{Y}\ F)))) \end{array}$$

Carry out enough beta reductions for the remaining term $(\mathbf{Y}\, A\ (\mathbf{Y}\, F)\ (\text{cdr}\,(\mathbf{Y}\, F)))$ until you can see a couple of elements of this cons-list. Can you guess what $\mathbf{Y}\ F$ reduces to, from these observations?

Exercise 4: Church encodings

Recall the encoding of booleans in the lambda calculus:

$$\begin{array}{rcl} \textbf{true} & = & \lambda x.\, \lambda y.\, x \\ \textbf{false} & = & \lambda x.\, \lambda y.\, y \\ \textbf{if} & & \lambda x.\, x \end{array}$$

and the definition of the and function:

$$\text{and} = \lambda p.\, \lambda q.\, \textbf{if}\ p\ q\ \textbf{false}$$

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

a. and **true false**

b. and **false true**

c. and **true** (and **true false**)

d. and **false** (and **true false**)

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