

# TYPES AND $\lambda$ -CALCULUS

## Problem Sheet 4

This week, Questions 3 and 6 will be marked.

- \* 1. Justify each of the following conversions  $M \approx N$  by finding a common reduct  $P$ , i.e. such that  $M \triangleright^* P$  and  $N \triangleright^* P$ .
- (a)  $(\lambda x. x)y \approx (\lambda xy. x) y z$
  - (b)  $(\lambda x. M)N \approx M[N/x]$
  - (c)  $\text{fix } (\underline{\text{const } 1}) \approx (\lambda x. x \underline{2}) (\underline{\text{const } (\text{pred } 2)})$
  - (d)  $z (\underline{\text{const id div}}) \underline{\text{div}} \approx z \underline{\text{id}} (\underline{\text{const div id}})$

- \* 2. Define  $\underline{Y} = \lambda f. (\lambda x. f (x x))(\lambda x. f (x x))$ .

Show that  $\underline{Y}$  is also a fixed point combinator, i.e show that, for all terms  $M$ :

$$\underline{Y}M \approx M(\underline{Y}M)$$

- \*\* 3. Prove Lemma 8.1 of the notes, i.e. show all of the following:

**Reflexivity** For all  $M$ :  $M \approx M$ .

**Symmetry** For all  $M, N$ :  $M \approx N$  implies  $N \approx M$ .

**Transitivity** For all  $M, N$  and  $P$ :  $M \approx P$  and  $P \approx N$  implies  $M \approx N$ .

**Compatibility** For all  $M, N$  and  $C[\ ]$ : if  $M \approx N$  then  $C[M] \approx C[N]$ .

There is no need for any induction. For compatibility, you will need to use 9(b) from the previous problem sheet.

- \* 4. Following the recipe, addition add, may be defined as:

$$\text{fix } (\lambda f x y. \text{ifz } x \ y \ (S \ (f \ (\text{pred } x) \ y)))$$

Give a complete reduction sequence from add 2 3 to 5.

- \* 5.

- (a) Prove that add satisfies the following equation:

$$\text{add} \approx \lambda x y. \text{ifz } x \ y \ (S \ (\text{add} \ (\text{pred } x) \ y))$$

- (b) Prove that add satisfies the following equations. Induction is not necessary.

$$\text{add } \underline{0} \ \underline{m} \approx \underline{m} \quad \text{and} \quad \text{add } (\underline{n+1}) \ \underline{m} \approx S \ (\text{add } \underline{n} \ \underline{m})$$

Hint: In practice, you nearly always want to replace an occurrence of add with the right-hand-side of the equation in (a), rather than by its actual definition (and the same can be said for any recursive function defined using “the recipe”).

- \*\* 6. Use fix to define the recursive triangular number function: using the "recipe", give a combinator Tri that satisfies:

$$\text{Tri } \underline{0} \approx \underline{0} \quad \text{and} \quad \text{Tri } (\underline{n+1}) \approx \text{add } (\underline{n+1}) \ (\text{Tri } \underline{n})$$

Convince yourself that Tri 2  $\approx$  3 (this is obvious if you believe that your implementation of Tri really satisfies the given equations).

- \*\* 7. Define multiplication, i.e. construct a term mult that satisfies the following specification:

$$\text{mult } \underline{0} \ \underline{m} \approx \underline{0} \quad \text{and} \quad \text{mult } \underline{n+1} \ \underline{m} \approx \text{add } \underline{m} \ (\text{mult } \underline{n} \ \underline{m})$$

Convince yourself that mult 2 2  $\approx$  4.

\*\* 8. Prove that if  $M \approx N$  and  $N$  is a normal form, then  $M \triangleright^* N$ .

Therefore, we now know that e.g.  $\underline{\text{Tri}}\ \underline{2} \triangleright^* \underline{3}$  and  $\underline{\text{mult}}\ \underline{2}\ \underline{2} \triangleright^* \underline{4}$ , so these definitions actually *compute* an output given an input.