CIS 500 Software Foundations

Homework Assignment 2

Untyped Lambda Calculus

Due: Monday, September 26, 2005, by noon

Submission instructions:

You must submit your solutions electronically (in ascii, postscript, or PDF format). Electronic solutions should be submitted following the same instructions as last time; these can be found at `http://www.seas.upenn.edu/~cis500/homework.html`. Do **not** email your solutions to us.

1 Exercise Show the derivation of one single step for each of these lambda calculus terms:

1. $(\lambda \mathtt{t}.\lambda \mathtt{f}.\mathtt{t})(\lambda \mathtt{t}.\lambda \mathtt{f}.\mathtt{f})(\lambda \mathtt{x}.\mathtt{x})$
2. $(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})$
3. $(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}\ \mathtt{x})(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}\ \mathtt{x})$

2 Exercise Write down the normal forms for the following lambda calculus terms if a normal form exists:

1. $(\lambda \mathtt{t}.\lambda \mathtt{f}.\mathtt{t})(\lambda \mathtt{t}.\lambda \mathtt{f}.\mathtt{f})(\lambda \mathtt{x}.\mathtt{x})$
2. $(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})(\lambda \mathtt{x}.\mathtt{x})$
3. $(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}\ \mathtt{x})(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}\ \mathtt{x})$
4. $(\lambda \mathtt{y}.(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x})(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}))$
5. $(\lambda \mathtt{y}.(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x})\ \mathtt{y})(\lambda \mathtt{x}.\mathtt{x})$

3 Exercise These problems involve programming in the untyped lambda calculus. You can test your solutions using the lambda calculus interpreter installed on `eniac-l.seas.upenn.edu`. The interpreter may be run with: ∼`cis500/fulluntyped/f filename.f` where `filename.f` is a file containing lambda calculus expressions. The output is the value of evaluating those expressions (if any). The syntax can been seen by looking at the examples in the file `test.f` in ∼`cis500/fulluntyped`.

1. TAPL 5.2.4.
2. TAPL 5.2.7.
3. TAPL 5.2.8.
4. Define a lambda term that takes two Church booleans and returns the logical "nand" of `b` and `c` (i.e. "not and").
5. Write an `isone` term that takes a Church numeral and returns `tru` if it is equivalent to $\mathtt{c}_1$ and `fls` otherwise. Do not use the function `equal` that you defined for TAPL 5.2.7.
6. Using the encoding of lists from 5.2.8, write a function that determines if any of the boolean values in a list are `tru` without using `fix`.
7. Now write a version of the function that determines if any of the boolean values in a list are `tru` that does use `fix`.

4 Exercise Questions about scope:

1. Which of these terms are closed?

   (a) $\lambda \mathtt{z}.\mathtt{z}\ \mathtt{z}$

   (b) $\lambda \mathtt{w}.\mathtt{x}$

   (c) $(\lambda \mathtt{m}.\mathtt{m})\ \mathtt{m}$

   (d) $\lambda \mathtt{z}.\lambda \mathtt{y}.\mathtt{z}\ \mathtt{y}$

2. Which of these pairs of terms are alpha-equivalent?

$\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{x}\ \mathtt{y}$	$\lambda \mathtt{x}.\lambda \mathtt{x}.\mathtt{x}\ \mathtt{x}$
$\lambda \mathtt{w}.\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{w}\ (\mathtt{x}\ \mathtt{y})$	$\lambda \mathtt{x}.\lambda \mathtt{y}.\lambda \mathtt{w}.\mathtt{w}\ (\mathtt{x}\ \mathtt{y})$
$\mathtt{x}$	$\mathtt{y}$
$\lambda \mathtt{w}.(\lambda \mathtt{x}.\mathtt{x})\ \mathtt{x}$	$\lambda \mathtt{v}.(\lambda \mathtt{y}.\mathtt{y})\ \mathtt{y}$
$\lambda \mathtt{w}.\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{w}\ (\mathtt{x}\ \mathtt{y})$	$\lambda \mathtt{x}.\lambda \mathtt{y}.\lambda \mathtt{w}.\mathtt{x}\ (\mathtt{y}\ \mathtt{w})$

5 Exercise TAPL 5.3.8. Additionally, show that the big step semantics is equivalent to the small step semantics.

6 Exercise Suppose we add a new non-value term `wrong` to the syntax of the untyped lambda calculus, and add the following rules to the big-step semantics for lambda calculus terms (that you defined in the previous exercise).

$$\frac{}{\mathtt{x} \Downarrow \mathtt{wrong}}\ \text{B-VarWrong} \qquad \frac{\mathtt{t}_1 \Downarrow \mathtt{wrong}}{\mathtt{t}_1\ \mathtt{t}_2 \Downarrow \mathtt{wrong}}\ \text{B-App1Wrong}$$

$$\frac{\mathtt{t}_1 \Downarrow \lambda \mathtt{x}.\mathtt{t}_{12} \quad \mathtt{t}_2 \Downarrow \mathtt{wrong}}{\mathtt{t}_1\ \mathtt{t}_2 \Downarrow \mathtt{wrong}}\ \text{B-App2Wrong} \qquad \frac{\mathtt{t}_1 \Downarrow \lambda \mathtt{x}.\mathtt{t}_{12} \quad \mathtt{t}_2 \Downarrow \mathtt{v}_2 \quad [\mathtt{x} \mapsto \mathtt{v}_2]\mathtt{t}_{12} \Downarrow \mathtt{wrong}}{\mathtt{t}_1\ \mathtt{t}_2 \Downarrow \mathtt{wrong}}\ \text{B-App3Wrong}$$

Consider the following theorem, that states that the evaluation of closed terms "doesn't go wrong".

> if $FV(\mathtt{t}) = \emptyset$ and $\mathtt{t} \Downarrow \mathtt{t}'$ then $\mathtt{t}'$ is not `wrong`.

1. Give a derivation of $(\lambda \mathtt{y}.\mathtt{y}\ \mathtt{z})(\lambda \mathtt{x}.\mathtt{x}) \Downarrow \mathtt{wrong}$.
2. Why can we not prove the theorem by structural induction on $\mathtt{t}$?
3. What is wrong with stating the theorem as: if $FV(\mathtt{t}) = \emptyset$ then $\mathtt{t} \Downarrow \mathtt{v}$?
4. Prove the theorem by induction on the derivation of $\mathtt{t} \Downarrow \mathtt{t}'$.

7 Debriefing

1. How many hours did you spend on this assignment?
2. Would you rate it as easy, moderate, or difficult?
3. Did everyone in your study group participate?
4. How deeply do you feel you understand the material it covers (0%–100%)?

If you have any other comments, we would like to hear them; please send them `cis500@cis.upenn.edu`.