IIT CS440: Programming Languages and Translators

Homework 5: Big-Step Semantics and Lambda Calculus

Prof. Stefan Muller
TA: Xincheng Yang

Out: Tuesday, Apr. 4
Due: Thursday, Apr. 13 11:59pm CST

This assignment contains 7 written tasks, for a total of 50 points.

0 Logistics and Submission - Important

1. Make sure you read and understand the collaboration policy on the course website.
2. There's no programming on this assignment. Submit typed or *neatly* handwritten and scanned answers on Blackboard.

1 Inference Rules

Task 1.1 (Written, 10 points).

Write inference rules for the judgments "X is a food" and "X is a sandwich" by converting the following informal English statements into inference rules (one rule per statement):

1. Peanut butter is a food
2. Jelly is a food
3. If X and Y are foods, then "X and Y" is a food.
4. If X is a food, then "X between bread" is a sandwich.
5. If X is a sandwich, X is food.

2 Big-step Semantics

Task 2.1 (Written, 12 points).

Using the rules from class for the IMP language, prove that

$$\langle \{x \mapsto 0\}, \mathsf{if}\ x = 0\ \mathsf{then}\ x := x + \overline{1}\ \mathsf{else}\ \mathsf{skip}\rangle \Downarrow \{x \mapsto 1\}$$

You can use σ instead of $\{x \mapsto 0\}$.

3 λ Calculus

Task 3.1 (Written, 8 points).

For each of the following pairs of terms, say whether they're α-equivalent or not. If not, explain why in a sentence or two.

(a) $\lambda x.\lambda y.x\ y \stackrel{?}{\equiv} \lambda y.\lambda x.x\ y$

(b) $\lambda x.\lambda y.x\ b \stackrel{?}{\equiv} \lambda a.\lambda b.a\ z$

(c) $\lambda x.\lambda y.x\ y \stackrel{?}{\equiv} \lambda y.\lambda x.y\ x$

(d) $(\lambda x.x)\ (\lambda x.x) \stackrel{?}{\equiv} (\lambda x.x)\ (\lambda y.y)$

Task 3.2 (Written, 10 points).

Perform the following substitutions. You may need to α-convert to avoid capture. Just do the substitution, you don't need to reduce further.

(a) $[w/x](\lambda x.x\ y)\ (\lambda z.z)$

(b) $[w/x](f\ x\ ((\lambda x.x)\ z))$

(c) $[\lambda x.y/f]((\lambda y.f\ y)\ z)$

(d) $[\lambda x.x/f](\lambda x.f\ x)$

(e) $[\lambda x.z/f](\lambda z.f\ z)$

Task 3.3 (Written, 10 points).

Reduce the following term to a normal form in two different ways. You can use α conversions, β reductions or η reductions. For each reduction, show each step.

$$(\lambda x.\lambda y.x\ x\ y)\ (\lambda f.\lambda y.f\ y)\ ((\lambda y.y)\ (\lambda z.z))$$

Bonus Task 3.4 (Written, 2 points).

A language has the "Church-Rosser property" if, for any program in the language, any set of reductions results in the same normal form. As we've discussed, the lambda calculus has the Church-Rosser property. OCaml does not. Give an OCaml program that can be evaluated in two different ways (e.g., using call-by-name evaluation and call-by-value execution) with two different results. Explain briefly the two different evaluation strategies and the result of each.

Hint: All purely functional languages have the Church-Rosser property, so your example will need to use some non-functional feature of OCaml, such as `ref` or `Printf.printf`. You can consider two evaluations of a program to have different results if they print different text.

4 Standard Final Questions

Task 4.1 (Written, 0 points).

How long (approximately, in hours/minutes of actual working time) did you spend on this homework, total? Your honest feedback will help us with future homeworks.

Task 4.2 (Written, 0 points).

Who, if anyone, did you collaborate with (and in what way), and what outside sources, if any, did you consult in working on this homework?