Programming Languages 2

Homework 3 – WS 18

Tübingen, 8. November 2018

In order to be admitted to the exam, you have to successfully submit your homework every week, except for 2 weeks. A successful submission is one where you get at least 1 point.

Handin Please submit this homework until Thursday, November 15, either via email to Philipp Schuster (philipp.schuster@uni-tuebingen.de) before 12:00, or on paper at the beginning of the lab.

Groups You can work in groups of up to 2 people. Please include the names and Matrikelnummern of all group members in your submission.

Points For each of the Tasks you get between 0 and 2 points for a total of 6 points. You get:
1 point, if your submission shows that you tried to solve the task.
2 points, if your submission is mostly correct.

Task 1: Evaluation

Reduce the following terms until they reach a normal form. Use the reduction relation from the lecture.

1. \((\lambda a. \lambda b. a) \; (\lambda x. \; x)\)
2. \((\lambda a. \lambda b. a \; b) \; (\lambda x. \; \lambda y. \; x \; y) \; (\lambda z. \; z)\)
3. \((\lambda z. \; z \; z) \; (\lambda f. \; f \; (\lambda a. \; a))\)

You do not have to draw a derivation tree for the reduction relation, but you do have to write down all reduction steps. So for example for the term \((\lambda f. \; f \; (\lambda x. \; x)) \; (\lambda x. \; x)\) you would write down the following:

\((\lambda f. \; f \; (\lambda x. \; x)) \; (\lambda x. \; x) \rightarrow (\lambda x. \; x) \; (\lambda x. \; x) \rightarrow (\lambda x. \; x)\)
**Task 2: Church encoding**

We are going to encode lists in lambda calculus as folds, similarly to how in the lecture we encoded natural numbers as folds. The basic list functions look like this:

- \( \text{nil} = \lambda f. \lambda z. \, z \)
- \( \text{singleton} = \lambda x. \lambda f. \lambda z. \, f \, x \, z \)
- \( \text{cons} = \lambda h. \lambda t. \lambda f. \lambda z. \, f \, h \, (t \, f \, z) \)
- \( \text{fold} = \lambda f. \lambda z. \lambda l. \, l \, f \, z \)

Using this encoding, the list with elements \( c_1, c_2, c_3 \) would for example look like this:

\[
\text{onetwothree} = \lambda f. \lambda z. \, f \, (c_1 \, (f \, c_2 \, (f \, c_3 \, z)))
\]

Write the following functions, operating on this encoding of lists:

1. sum
2. map
3. length

You can use the macros that were defined in the lecture, especially helpful are \( \text{plus}, \, c_0, \, c_1, \, c_2, \ldots \)

**Task 3: Monotonicity**

Let the reduction relation \( \rightarrow \) and the function \( \text{size} \) for terms in untyped lambda calculus be defined as in the lecture. Prove or disprove: if \( t \rightarrow t' \), then \( \text{size}(t) > \text{size}(t') \).