Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Lukas Stevens

Lambda Calculus
Winter Term 2021/22
Solutions to Exercise Sheet 6

Exercise 1 (Reduction Relation with Closures)

For the evaluation of lambda terms that is closer to evaluation of programs in functional programming languages, one usually replaces textual substitution $t[v/x]$ with a more lazy approach that records the binding $x \mapsto v$ in an environment. These bindings are used whenever we need the value of a variable v.

In this approach abstractions $\lambda x.\ t$ do not evaluate to themselves, but to a pair $(\lambda x.\ t)[e]$, where e is the current environment. We call such pairs function *closures*.

a) Define a big-step reduction relation for the lambda calculus with function closures and environments.

b) Add explicit error handling for the case where the binding of a free variable v cannot be found in the enviroment. Introduce an explicit value **abort** to indicate such an exception in the relation.

Solution

a) First recall the standard $\Rightarrow_{cbv}$ relation:

$$\lambda x.\ t \Rightarrow_{cbv} \lambda x.\ t$$

$$\frac{s \Rightarrow_{cbv} \lambda x.\ s' \qquad t \Rightarrow_{cbv} v \qquad s'[v/x] \Rightarrow_{cbv} w}{s\ t \Rightarrow_{cbv} w}$$

Note that there are no rules for variables since the reduction relation only considers closed terms. Now we define the relation for lambda calculus with closures:

$$\frac{e(x) = v}{e \vdash x \Rightarrow_{cbv} v} \qquad e \vdash \lambda x.\ t \Rightarrow_{cbv} (\lambda x.\ v)[e]$$

$$\frac{e \vdash t_1 \Rightarrow_{cbv} (\lambda x.\ t)[e'] \qquad e \vdash t_2 \Rightarrow_{cbv} v' \qquad e' + (x \mapsto v') \vdash t \Rightarrow_{cbv} v}{e \vdash t_1\, t_2 \Rightarrow_{cbv} v}$$

In the following example empty closures for lambdas are omitted for better readability:

$$\dfrac{\dfrac{}{() \vdash (\lambda u.\ u) \Rightarrow_{cbv} (\lambda u.\ u)} \qquad \dfrac{\dfrac{}{() \vdash (\lambda x\ y.\ x) \Rightarrow_{cbv} (\lambda x\ y.\ x)}}{(x \mapsto (\lambda u.\ u)) \vdash (\lambda y.\ x) \Rightarrow_{cbv} (\lambda y.\ x)[x \mapsto (\lambda u.\ u)]}}{() \vdash (\lambda x\ y.\ x)\ (\lambda u.\ u) \Rightarrow_{cbv} (\lambda y.\ x)[x \mapsto (\lambda u.\ u)]}$$

$$\dfrac{\dfrac{}{() \vdash (\lambda v.\ v) \Rightarrow_{cbv} (\lambda v.\ v)} \qquad \dfrac{\dfrac{\text{See above}}{() \vdash (\lambda x\ y.\ x)\ (\lambda u.\ u) \Rightarrow_{cbv} (\lambda y.\ x)[x \mapsto (\lambda u.\ u)]} \quad e(x) = (\lambda u.\ u)}{(x \mapsto (\lambda u.\ u)) + (y \mapsto (\lambda v.\ v)) \vdash x \Rightarrow_{cbv} (\lambda u.\ u)}}{() \vdash ((\lambda x\ y.\ x)\ (\lambda u.\ u))\ (\lambda v.\ v) \Rightarrow_{cbv} (\lambda u.\ u)}$$

b) We just need to add rules to propagate errors, and modify the existing rules to ensure that no subexpression evaluates to **abort**.

$$\dfrac{x \notin e}{e \vdash x \Rightarrow_{cbv} \mathbf{abort}} \qquad \dfrac{e(x) = v}{e \vdash x \Rightarrow_{cbv} v} \qquad e \vdash \lambda x.\ t \Rightarrow_{cbv} (\lambda x.\ t)[e]$$

$$\dfrac{e \vdash t_2 \Rightarrow_{cbv} v' \qquad \begin{matrix} e \vdash t_1 \Rightarrow_{cbv} (\lambda x.\ t)[e'] \\ e' + (x \mapsto v') \vdash t \Rightarrow_{cbv} v \end{matrix} \qquad v' \neq \mathbf{abort}}{e \vdash t_1\, t_2 \Rightarrow_{cbv} v}$$

$$\dfrac{e \vdash t_1 \Rightarrow_{cbv} \mathbf{abort} \qquad e \vdash t_2 \Rightarrow_{cbv} v \qquad v \neq \mathbf{abort}}{e \vdash t_1\, t_2 \Rightarrow_{cbv} \mathbf{abort}} \qquad \dfrac{e \vdash t_2 \Rightarrow_{cbv} \mathbf{abort}}{e \vdash t_1\, t_2 \Rightarrow_{cbv} \mathbf{abort}}$$

Exercise 2 (Better Translation Algorithm)

Give a variant of the translation algorithm that produces shorter terms. More specifically, define a variant of $\lambda^* x.\ t$ that analyzes more precisely where x actually appears in t.

Solution

$$\begin{array}{rcll}
\lambda^* x.\ x & = & I & \\
\lambda^* x.\ X & = & \mathsf{K}\ X & \text{if } x \notin FV(X) \\
\lambda^* x.\ X\ x & = & X & \text{if } x \notin FV(X) \\
\lambda^* x.\ (X\ Y) & = & \mathsf{B}\ X\ (\lambda^* x.\ Y) & \text{if } x \notin FV(X) \wedge x \in FV(Y) \\
\lambda^* x.\ (Y\ X) & = & \mathsf{C}\ (\lambda^* x.\ Y)\ X & \text{if } x \notin FV(X) \wedge x \in FV(Y) \\
\lambda^* x.\ (X\ Y) & = & \mathsf{S}\ (\lambda^* x.\ X)\ (\lambda^* x.\ Y) & \text{if } x \in FV(X, Y)
\end{array}$$

where $\mathsf{B} := \mathsf{S}\ (\mathsf{K}\ \mathsf{S})\ \mathsf{K}$ and $\mathsf{C} := \mathsf{S}\ (\mathsf{B}\ \mathsf{B}\ \mathsf{S})\ (\mathsf{K}\ \mathsf{K})$. B and C fulfill the following properties

$$\mathsf{B}\ X\ Y\ Z \to^* X\ (Y\ Z)$$

$$\mathsf{C}\ X\ Y\ Z \to^* X\ Z\ Y$$

Homework 3 (Proofs with Small-steps and Big-steps)

Let $\omega := \lambda x.\ x\ x$ and

$$t\ :=\ (\lambda x.\ (\lambda x\ y.\ x)\ z\ y)\ (\omega\ \omega\ ((\lambda x\ y.\ x)\ y))\,.$$

Prove the following:

a) $t \Rightarrow_n z$

b) $t \rightarrow^3_{cbv} t$

c) $t \not\rightarrow^+_{cbn} t$

Homework 4 (More Combinators)

Find combinators O and W such that:

$$\mathsf{O}\ \rightarrow^+ \mathsf{O}$$
$$\mathsf{W}\,X\,Y \rightarrow^* X\,Y\,Y$$

Homework 5 (Mocking Birds)

Consider a combinatory logic that only provides the basic combinators B and M (the "mocking bird") where:

$$\mathsf{B}\ X\ Y\ Z \rightarrow^* X\ (Y\ Z)$$
$$\mathsf{M}\ X \rightarrow^* X\ X$$

Prove the following properties of this logic:

a) For every combinator X, there is a combinator Y such that $Y \rightarrow^* X\ Y$.

b) For all combinators U and W, there exist combinators X and Y such that $Y \rightarrow^* U\ X$ and $X \rightarrow^* W\ Y$.

Homework 6 (Correctness of the Translation Algorithm)

Show that the translation algorithm given in the tutorial is correct. That is, show that it fulfills the following property:

$$(\lambda^* x.\ X)\,Y \rightarrow^* X[Y/x]$$