Exercise 1 (Progress Property)

Let $t$ be a closed and well-typed term, i.e. $\vdash t : \tau$ for some $\tau$. Show that $t$ is either a value or there is a $t'$ such that $t \rightarrow_{cbv} t'$.

Solution

The proof follows an induction on the derivation of $\vdash t : \tau$. The variable case cannot occur. Abstractions are values, so there is nothing to do here. For the application case, assume $t = t_1 t_2$ and $\vdash t_1 : \tau_1 \rightarrow \tau_2$ and $\vdash t_2 : \tau_1$. By the induction hypothesis, both, $t_1$ and $t_2$, can take a step or are a value. If $t_1$ can take a step, we can use the left application rule on $t$. If $t_1$ is a value and $t_2$ can take a step, then the right application rule can be used. If $t_1$ and $t_2$ are both values, we know $t_1 = \lambda x. t'_1$ for some $t'_1$ as $t_1$ is of type $\tau_1 \rightarrow \tau_2$. Thus we can apply the rule for reducing an abstraction.

Exercise 2 (Normal Form)

Show that every type-correct $\lambda\to$-term has a $\beta$-normal form.

Solution

We prove the statement by coming up with a terminating reduction relation $\rightarrow_p$ that, when repeatedly applied, reduces a given term to $\beta$-normal form. Furthermore, we define a well-founded order $<_T$ on terms and show that $t_1 \rightarrow_p t_2$ implies $t_1 <_T t_2$. By induction on $<_T$ it then follows that $\rightarrow_p$ is terminating.

The reduction strategy is chosen such that it decreases the types of subterms.

Let $|\tau|$ be the size of a type $\tau$, i.e. the number of function-arrows occurring in $\tau$.

$$
|\alpha| = 0 \\
|\alpha \to \beta| = |\alpha| + |\beta| + 1
$$

With this measure, we can assign a natural number to each $\beta$-redex:

$$(\lambda x. s) t |^\Gamma = |\tau_1 \rightarrow \tau_2| \quad \text{where} \quad \Gamma \vdash (\lambda x. s) : \tau_1 \rightarrow \tau_2
$$

We assign each term $t$ a multiset $M_t$. In order to account for the potentially non-empty environment $\Gamma$ in the subterms of $t$, we first define $M_t^\Gamma$ recursively and then set $M_t := M_t^\Gamma$.

\[
\begin{align*}
t = u v & \implies M_t^\Gamma = M_u^\Gamma \cup M_v^\Gamma \cup \{| u v |^\Gamma \}. \text{ } u v \text{ is a } \beta\text{-redex} \\
t = (\lambda x. s) & \implies \Gamma \vdash (\lambda x. s) : \tau_1 \rightarrow \tau_2 \implies M_t^\Gamma = M_{s|^{\tau_1}}^\Gamma \\
t = x & \implies M_t = \{\}
\end{align*}
\]
We can view multisets as functions into the natural numbers and define an ordering on them:

\[ M <_M N \iff M \neq N \land (\forall y. M(y) > N(y) \implies (\exists x. y < x \land M(x) < N(x))) \]

It can be proved that the multiset ordering terminates (is well-founded). The ordering naturally extends to terms, i.e. \( u <_T v \iff M_u <_M M_v \).

If one regards a \( \beta \)-redex of the form \( r = (\lambda x. u) v \) with \( \Gamma \vdash r : \tau \) and \( u \) and \( v \) in \( \beta \)-NF, then we have \( M_r >_M M_{r'} \) for the reduct \( r' = u[v/x] \).

This is because although the substitution may create new \( \beta \)-redexes \( w \), we have \( |w| < |r| \) for all those \( w \) in \( r' \):

- Note that \( \Gamma \vdash (\lambda x. u) : \tau_1 \to \tau_2 \) and \( \Gamma \vdash v : \tau_2 \) for some \( \tau_1, \tau_2 \) must hold.
- Since \( v \in \text{NF} \), \( w \) is of the form \( (v v') \) with \( v = (\lambda x. s) \) for some \( s \) and thus \( |w| = |\tau_1| < |\tau_1 \to \tau_2| = |r| \).

Thus, if we choose a reduction strategy \( \to_{p} \) that reduces an innermost \( \beta \)-redex in \( t \), we have:

\[ t \to_{p} t' \implies M_t >_M M_{t'} \]

We can obtain such a reduction strategy by restricting the first rule of \( \to_{\beta} \) to:

\[ \frac{s \in \text{NF} \quad t \in \text{NF}}{(\lambda x. s) t \to_{p} s[t/x]} \]

As the multiset ordering terminates, also the chosen reduction strategy must terminate. If it terminates with \( t' \), then \( t' \) is in \( \beta \)-NF because otherwise \( t' \) would contain a regex and therefore an innermost regex that can be reduces with \( \to_{p} \).

\[ \square \]
Homework 3 (Typing)

a) Prove:

\[ [\cdot] \vdash (\lambda x : \tau_2 \rightarrow \tau_3. \lambda y : \tau_1 \rightarrow \tau_2. \lambda z : \tau_1. \ x \ (y \ z)) : (\tau_2 \rightarrow \tau_3) \rightarrow (\tau_1 \rightarrow \tau_2) \rightarrow \tau_1 \rightarrow \tau_3 \]

b) Give suitable solutions for ?\tau_1, ?\tau_2, ?\tau_3 and ?\tau_4 and prove that the term is type-correct given your solution.

\[ [\cdot] \vdash \lambda x : ?\tau_1. \ \lambda y : ?\tau_2. \ \lambda z : ?\tau_3. \ x \ y \ (y \ z) : ?\tau_4 \]

Homework 4 (\(\beta\)-reduction preserves types)

A type system has the subject reduction property if evaluating an expression preserves its type. Prove that the simply typed \(\lambda\)-calculus \((\lambda^{\rightarrow})\) has the subject reduction property:

\[ \Gamma \vdash t : \tau \wedge t \rightarrow_{\beta} t' \implies \Gamma \vdash t' : \tau \]

Hints: Use induction over the inductive definition of \(\rightarrow_{\beta}\) (Def. 1.2.2). State your inductive hypotheses precisely – it may help to introduce a binary predicate \(P(t, t')\) to express the property you are proving by induction. Also note that the proof will require rule inversion: Given \(\Gamma \vdash t : \tau\), the shape of \(t\) (variable, application, or \(\lambda\)-abstraction) may determine which typing rule must have been used to derive the typing judgment.

Within your proof, you are free to use the following lemma about substitution:

\[ \Gamma \vdash u : \tau_0 \wedge \Gamma[x : \tau_0] \vdash t : \tau \implies \Gamma \vdash t[u/x] : \tau \] (1)

Homework 5 (Implementation of multiset-ordering and reduction)

Implement the multiset ordering and the reduction strategy from the second tutorial exercise in your favorite programming language.