TD: Operational Semantics

Full $\beta$-reduction is the basis for the symbolic manipulation of $\lambda$-terms, e.g., in proof assistants, in program transformations, and in higher-order unification. However, when the $\lambda$-calculus is regarded as the core of a programming language it is sensible to consider weaker reduction strategies.

A weak reduction is a strategy to reduce $\lambda$-terms that does not reduce under abstractions (i.e., in the body of a function).

**Memo:**
A one-step $\beta$-reduction is the closure of the $\beta$-rule under context.

\[ C ::= [ ] | Cu | uC \mid \lambda x.C \]

**Said otherwise**
A one-step $\beta$-reduction is given inductively by

\[
\begin{align*}
(\lambda x.f) \ u & \rightarrow^\beta t[x/u] \\
\lambda x.t & \rightarrow^\beta \lambda x.f' \\
t & \rightarrow^\beta t' \\
u & \rightarrow^\beta u' \\
t \ u & \rightarrow^\beta t' \ u
\end{align*}
\]

**Ex. 0.** Change the rules above, so that we do not reduce under $\lambda$.
Evaluate $\text{K(II)}$.

**Def.** A value $V$ is a closed $\lambda$-term of the shape $\lambda x.t$ (a $\lambda$-abstraction).

\[
E ::= [ ] \mid E t \quad \text{(call-by-name evaluation context)}
\]
\[
E ::= [ ] \mid E t \mid V E \quad \text{(call-by-value evaluation context)}
\]

<table>
<thead>
<tr>
<th>Call-by-value (weak)</th>
<th>Call-by-name (weak)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\lambda x.f) \ V \rightarrow_v t[V/x]$</td>
<td>$(\lambda x.t) \ u \rightarrow_n t[u/x]$</td>
</tr>
<tr>
<td>$t \rightarrow_v t'$</td>
<td>$u \rightarrow_v u'$</td>
</tr>
<tr>
<td>$u \rightarrow_v u'$</td>
<td>$V \rightarrow_v V V'$</td>
</tr>
<tr>
<td>$t \rightarrow_n t'$</td>
<td>$t \rightarrow_n t' \ u$</td>
</tr>
</tbody>
</table>

What are the normal forms? What are the normal forms?

**Ex. 1.** Progression.
Assume $t$ is a closed term. Prove that (in both systems) values are exactly the normal forms.
(Hence if a term is not a value, it has a reduction step)

**Ex. 2.** Determinism

**Proposition (decomposition)** Let $t$ be a closed $\lambda$-term. Then either $t$ is a value or there is a unique call-by-name (call-by-value) evaluation context $E$ such that:

\[
t \equiv E[(\lambda x.u_1)u_2] \quad t \equiv E[(\lambda x.u_1)V].
\]

1. If $t \rightarrow_v u$ and $t \rightarrow_v u'$, then $u = u'$.
2. If $t \rightarrow_n u$ and $t \rightarrow_n u'$, then $u = u'$.
3. If $t \not\downarrow_v V$ and $t \not\downarrow_v V'$, then $V = V'$.
4. If $t \not\downarrow_n V$ and $t \not\downarrow_n V'$, then $V = V'$. 
<table>
<thead>
<tr>
<th>Call-by-value (small-steps)</th>
<th>Call-by-value (BIG-STEPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[(\lambda x.t) V \leadsto_t t[V/x]]</td>
<td>[t \Downarrow_r \lambda x.r \quad u \Downarrow_t, W \quad r[W/x] \Downarrow_t, V]</td>
</tr>
<tr>
<td>[t \leadsto_r t' \quad u \leadsto_s u']</td>
<td>[t \Downarrow_t \quad V \quad V \Downarrow_t \quad V]</td>
</tr>
<tr>
<td>[tu \leadsto_r t'u]</td>
<td>[tu \Downarrow_t \quad V \Downarrow_t \quad V]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Call-by-name (small-steps)</th>
<th>Call-by-name (BIG-STEPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[(\lambda x.t) u \leadsto_n t[u/x]]</td>
<td>[t \Downarrow_n \lambda x.r \quad r[u/x] \Downarrow_n V]</td>
</tr>
<tr>
<td>[t \leadsto_n t']</td>
<td>[t \Downarrow_n \lambda x.r \quad r[u/x] \Downarrow_n V]</td>
</tr>
<tr>
<td>[tu \leadsto_n t'u]</td>
<td>[tu \Downarrow_n V]</td>
</tr>
</tbody>
</table>

**EX 3. Relating big steps and small steps**

1. If \(t \Downarrow_r V\), then \(t \leadsto_r^* V\).
2. If \(t \Downarrow_n V\), then \(t \leadsto_n^* V\).
3. If \(t \leadsto_n^* u\) and \(u\) is a CBV normal-form, then \(t \Downarrow_n u\).
4. If \(t \leadsto_n^* u\) and \(u\) is a CBN normal-form, then \(t \Downarrow_n u\).

**EX 4. Remember Church numerals?**

Consider the definition of Church Numerals:

\[\bar{n} \equiv \lambda f.\lambda x. (f \ldots (f x) \ldots)\]

and the following encodings:

\[S \equiv \lambda n.\lambda f.\lambda x.f(n fx)\] (successor)

1. Evaluate: \(S 1\)

2. Does the following statement still make sense in CBV?

Think of the Church numeral \(\bar{n}\) as the procedure that takes a function-input and an argument-input, and applies the function \(n\)-times to the argument.

**EX 5. Scott Numerals**

\[\begin{align*}
\bar{0} & \equiv k \\
\bar{n} + 1 & \equiv \lambda xy. y^n 1x \\
\text{succ} & \equiv \lambda nxy.yn \\
\text{pred} & \equiv \lambda p. p \theta i \\
\text{case} & \equiv \lambda xyz. xyz
\end{align*}\]

where \(\theta\) is any closed term and \(i\) the identity.

Evaluate:

1. succ [2]
2. pred [2]
3. case [n] fg (for f and g values)