Semantics of Programming Languages

An abstract machine for evaluating closed terms of the untyped lambda calculus has configurations which are non-empty lists of closed terms. Its transitions are of two forms:

\[(\text{app})\] \[(M_1 M_2) :: L \rightarrow M_1 :: M_2 :: L\]

\[(\text{abs})\] \[\lambda x (M_1) :: M_2 :: L \rightarrow M_1[M_2/x] :: L\]

where :: denotes list concatenation and \(M_1[M_2/x]\) denotes the result of substituting \(M_2\) for all free occurrences of the variable \(x\) in \(M_1\). Let \(\downarrow\) be the binary relation between closed terms inductively defined by the following axioms and rules:

\[(\downarrow_{\text{abs}})\] \[\lambda x (M) \downarrow \lambda x (M)\]

\[(\downarrow_{\text{app}})\] \[M_1 \downarrow \lambda x (M_2) \quad M_2[M_3/x] \downarrow \lambda x (M_4)\]

\[M_1 M_3 \downarrow \lambda x (M_4)\]

(a) Prove by Rule Induction that if \(M_1 \downarrow \lambda x (M_2)\) holds, then so does \(M_1 :: L \rightarrow^* \lambda x M_2 :: L\), where \(\rightarrow^*\) denotes the reflexive-transitive closure of the transition relation \(\rightarrow\). \([5 \text{ marks}]\)

(b) Prove by Mathematical Induction on \(n\) that if \((\ldots(M[M_0/x] M_1) M_2) \ldots)M_n \downarrow \lambda x (M')\), then \((\ldots(((\lambda x (M)) M_0) M_1) M_2) \ldots)M_n \downarrow \lambda x (M')\). \([5 \text{ marks}]\)

(c) Given a configuration \(M :: L\), let \(M@L\) denote the closed term defined by induction on the length of the list \(L\) by:

\[M@\text{nil} \overset{\text{def}}{=} M\] and \(M@(M' :: L) \overset{\text{def}}{=} (M M')@L\). Using (b), show by case analysis for \(\rightarrow\) that if \(M_1 :: L_1 \rightarrow M_2 :: L_2\) and \(M_2@L_2 \downarrow \lambda x (M')\) hold, then so does \(M_1@L_1 \downarrow \lambda x (M')\). \([5 \text{ marks}]\)

(d) Deduce from (a) and (c) that \(M_1 \downarrow \lambda x (M_2)\) holds if and only if \(M_1 :: \text{nil} \rightarrow^* \lambda x (M_2) :: \text{nil}\) does. \([5 \text{ marks}]\)