Free Variables and Substitution for MinML

Robert Harper

extracted & edited by A. Appel, 2001

We may define the set of *free variables* of an expression e, $\mathrm{FV}(e)$, as follows:

$$
\begin{array}{rcl}
\mathrm{FV}(x) & = & \{\, x \,\} \\
\mathrm{FV}(n) & = & \emptyset \\
\mathrm{FV}(\texttt{true}) & = & \emptyset \\
\mathrm{FV}(\texttt{false}) & = & \emptyset \\
\mathrm{FV}(o(e_1, \ldots, e_n)) & = & \mathrm{FV}(e_1) \cup \cdots \cup \mathrm{FV}(e_n) \\
\mathrm{FV}(\texttt{if}\, e \,\texttt{then}\, e_1 \,\texttt{else}\, e_2 \,\texttt{fi}) & = & \mathrm{FV}(e) \cup \mathrm{FV}(e_1) \cup \mathrm{FV}(e_2) \\
\mathrm{FV}(\texttt{fun}\, f\, \texttt{(}x\texttt{:}\tau_1\texttt{)}\texttt{:}\tau_2 \,\texttt{is}\, e \,\texttt{end}) & = & \mathrm{FV}(e) \backslash \{\, f, x \,\} \\
\mathrm{FV}(\texttt{apply}(e_1, e_2)) & = & \mathrm{FV}(e_1) \cup \mathrm{FV}(e_2)
\end{array}
$$

We say that the variable x is *free* in the expression e iff $x \in \mathrm{FV}(e)$. An expression e is *closed* iff $\mathrm{FV}(e) = \emptyset$; that is, a closed expression has no free variables.

Capture-avoiding substitution of an expression e for free occurrences of a variable x in another expression e', written $[e/x]e'$, is (partially) defined as follows:

$$
\begin{array}{rcl}
[e/x]x & = & e \\
[e/x]n & = & n \\
[e/x]\texttt{true} & = & \texttt{true} \\
[e/x]\texttt{false} & = & \texttt{false} \\
[e/x]o(e_1, \ldots, e_n) & = & o([e/x]e_1, \ldots, [e/x]e_n) \\
[e/x]\texttt{if}\, e \,\texttt{then}\, e_1 \,\texttt{else}\, e_2 \,\texttt{fi} & = & \texttt{if}\, [e/x]e \,\texttt{then}\, [e/x]e_1 \,\texttt{else}\, [e/x]e_2 \,\texttt{fi} \\
[e/x]\texttt{fun}\, f\, \texttt{(}y\texttt{:}\tau_1\texttt{)}\texttt{:}\tau_2 \,\texttt{is}\, e'' \,\texttt{end} & = & \texttt{fun}\, f\, \texttt{(}y\texttt{:}\tau_1\texttt{)}\texttt{:}\tau_2 \,\texttt{is}\, e'' \,\texttt{end} \quad \textit{if } x = f \textit{ or } x = y \\
[e/x]\texttt{fun}\, f\, \texttt{(}y\texttt{:}\tau_1\texttt{)}\texttt{:}\tau_2 \,\texttt{is}\, e'' \,\texttt{end} & = & \texttt{fun}\, f\, \texttt{(}y\texttt{:}\tau_1\texttt{)}\texttt{:}\tau_2 \,\texttt{is}\, [e/x]e'' \,\texttt{end} \quad \textit{if } \{\, f, y \,\} \cap (\mathrm{FV}(e) \cup \{\, x \,\}) = \emptyset \\
[e/x]\texttt{apply}(e_1, e_2) & = & \texttt{apply}([e/x]e_1, [e/x]e_2)
\end{array}
$$

Simultaneous capture-avoiding substitution, written $[e_1, \ldots, e_n/x_1, \ldots, x_n]e$, is defined in an analogous manner.

Capture-avoiding substitution is *undefined* if the condition in the penultimate equation is not met! In this case free occurrences of f or y in e would be *captured* by the binder for f and y, thereby erroneously changing the meanings of the “pronouns”. This means, for example, that

$$[x/y]\texttt{fun}\, f\, \texttt{(}x\texttt{:int)}\texttt{:int}\,\texttt{is}\, x + y \,\texttt{end}$$

is *undefined*, rather than equal to

$$\texttt{fun}\, f\, \texttt{(}x\texttt{:int)}\texttt{:int}\,\texttt{is}\, x + x \,\texttt{end},$$

wherein capture of x has occurred.

The possibility of capture during substitution can always be avoided by *renaming of bound variables*.