15-399 Supplementary Notes: Substitution

Robert Harper

February 17, 2005

Substitution

We will make frequent use of *substitution* of a term, M, for the free occurrences of a variable, u, in another term, N. This is written $[M/u]N$. The intuitive idea is clear, but there are pitfalls that must be avoided; some examples are given in the Pfenning notes. Here we give a precise definition of substitution to have as a reference.

Free and Bound Variables

First we define the set $\mathrm{FV}(M)$ of *free variables* occurring in M.

$$
\begin{array}{rcl}
\mathrm{FV}(u) & = & \{\, u \,\} \\
\mathrm{FV}(\langle\,\rangle) & = & \emptyset \\
\mathrm{FV}(\langle M_1, M_2\rangle) & = & \mathrm{FV}(M_1) \cup \mathrm{FV}(M_2) \\
\mathrm{FV}(\mathbf{fst}(M)) & = & \mathrm{FV}(M) \\
\mathrm{FV}(\mathbf{snd}(M)) & = & \mathrm{FV}(M) \\
\mathrm{FV}(\lambda u{:}P.M) & = & \mathrm{FV}(M) \setminus \{\, u \,\} \\
\mathrm{FV}(M_1\, M_2) & = & \mathrm{FV}(M_1) \cup \mathrm{FV}(M_2) \\
\mathrm{FV}(\mathbf{abort}_P(M)) & = & \mathrm{FV}(M) \\
\mathrm{FV}(\mathbf{inl}(M)) & = & \mathrm{FV}(M) \\
\mathrm{FV}(\mathbf{inr}(M)) & = & \mathrm{FV}(M) \\
\mathrm{FV}(\mathbf{case}\, M \;\mathbf{of}\; \mathbf{inl}(u_1) \Rightarrow M_1 \mid \mathbf{inr}(u_2) \Rightarrow M_2) & = & \\
\multicolumn{3}{r}{\mathrm{FV}(M) \cup (\mathrm{FV}(M_1) \setminus \{\, u_1 \,\}) \cup (\mathrm{FV}(M_2) \setminus \{\, u_2 \,\})}
\end{array}
$$

If a variable u occurs in M, but $u \notin \mathrm{FV}(M)$, then we say that u is *bound* in M. We say that M *lies apart* from N, written $M \# N$, iff $\mathrm{FV}(M) \cap \mathrm{FV}(N) = \emptyset$. We most often use this notation in the form $u \# M$, which therefore means $u \notin \mathrm{FV}(M)$. In particular, $u \# v$ iff $u \neq v$.

Renaming of Bound Variables

Intuitively, the terms $\lambda u{:}P.u$ and $\lambda v{:}P.v$ are the "same", since they differ only in the name of the bound variable. Two terms that differ only in the names of their bound variables are said to be *α-equivalent*, or *α-convertible*.[1] It is remarkably tricky to give a precise definition of this relation.

We first define the notion of *variable swapping*. We write $[u{\leftrightarrow}v]M$ to mean that *all* occurrences of u in M are to be replaced by v, and *all* occurrences of v in M are to be replaced by u. We emphasize "all", because the replacement applies even to binders such as λ. For example, $[u{\leftrightarrow}v]\lambda v{:}P.u = \lambda u{:}P.v$.

The relation $M =_\alpha N$ is inductively defined by the following rules:[2]

$$\frac{}{u =_\alpha u} \qquad \frac{}{\langle\rangle =_\alpha \langle\rangle} \qquad \frac{M =_\alpha M' \quad N =_\alpha N'}{\langle M, N\rangle =_\alpha \langle M', N'\rangle}$$

$$\frac{M =_\alpha M'}{\mathbf{fst}(M) =_\alpha \mathbf{fst}(M')} \qquad \frac{M =_\alpha M'}{\mathbf{snd}(M) =_\alpha \mathbf{snd}(M')}$$

$$\frac{M =_\alpha M'}{\lambda u{:}P.M =_\alpha \lambda u{:}P.M'} \qquad \frac{u \# v \quad v \# M \quad [u{\leftrightarrow}v]M =_\alpha M'}{\lambda u{:}P.M =_\alpha \lambda v{:}P.M'}$$

The last two lines are the most interesting. When comparing two λ's that bind the same variable, we simply compare their bodies. If, however, they have different bound variables, then we replace u by v (and v by u) in the first to reconcile the difference, and continue comparing. Since $v \# M$ the replacement of v by u in M does not change any *free* variable, but ensures that no confusion can occur when replacing u by v in M due to *bound* occurrences of v in M.

[1] The origin of the phrase is essentially a historical accident, but this terminology is too deeply entrenched to be changed now.

[2] We omit the rules for case analysis and abort for the sake of brevity.

Substitution

Substitution is inductively defined by the following clauses:

$$
\begin{array}{rcll}
[M/u]u & = & M & \\
[M/u]v & = & v & (u \# v) \\
\\
[M/u]\langle\,\rangle & = & \langle\,\rangle & \\
\\
[M/u]\langle N_1, N_2\rangle & = & \langle [M/u]N_1, [M/u]N_2\rangle & \\
[M/u]\mathbf{fst}(N) & = & \mathbf{fst}([M/u]N) & \\
[M/u]\mathbf{snd}(N) & = & \mathbf{snd}([M/u]N) & \\
\\
[M/u]\lambda v{:}P.N & = & \lambda v{:}P.[M/u]N & (v \# M) \\
[M/u](N_1\,N_2) & = & [M/u]N_1\,[M/u]N_2 & \\
\\
[M/u]\mathbf{abort}(N) & = & \mathbf{abort}([M/u]N) & \\
\\
[M/u]\mathbf{inl}(N) & = & \mathbf{inl}([M/u]N) & \\
[M/u]\mathbf{inr}(N) & = & \mathbf{inr}([M/u]N) &
\end{array}
$$

$$
\begin{array}{l}
[M/u]\mathbf{case}\,N\ \mathbf{of}\ \mathbf{inl}(u_1) \Rightarrow N_1 \mid \mathbf{inr}(u_2) \Rightarrow N_2 = \\
\mathbf{case}\,[M/u]N\ \mathbf{of}\ \mathbf{inl}(u_1) \Rightarrow [M/u]N_1 \mid \mathbf{inr}(u_2) \Rightarrow [M/u]N_2 \\
\hfill (u_1 \# \mathrm{FV}(M), u_2 \# \mathrm{FV}(M))
\end{array}
$$

The conditions on substitution into a λ or **case** expression mean that the substitution $[M/u]N$ need not be defined! For example, the attempted substitution $[\langle u, u\rangle/v]\lambda u{:}P.\langle u, v\rangle$ is undefined, because the bound variable, u, occurs free in $\langle u, u\rangle$. However, if we first rename the bound variable of the λ, then substitution is defined:

$$[\langle u, u\rangle/v]\lambda u'{:}P.\langle u', v\rangle = \lambda u'{:}P.\langle u', \langle u, u\rangle\rangle.$$

Similarly, the attempted substitution $[M/u]\lambda u{:}P.u$ is undefined, because the bound variable name is the same as the target of the substitution. But once again this is not a problem, because by renaming the bound variable to, say, v, where $v \neq u$, substitution is once again defined.

The undefinedness of substitution can always be avoided by renaming bound variables so as to ensure that the restrictions on substitution are met.

Theorem 0.1 *1. For any M, N, and u, there exists N' and N'' such that $N =_\alpha N'$ and $[M/u]N' = N''$.*

2. If $N =_\alpha N'$ and $N =_\alpha N''$ and $[M/u]N'$ and $[M/u]N''$ both exist, then $[M/u]N' =_\alpha [M/u]N''$.

Thus we say that *substitution is well-defined up to α-equivalence.*

Bound Variable Convention

Since bound variable names may be chosen arbitrarily, it is technically convenient to ignore the choice by systematically "modding out" by α-equivalence. This means that we *always* work with α-equivalence classes of terms, and implicitly choose representatives of each equivalence class so that all relevant substitutions are well-defined. This frees us from having to think about the fundamentally irrelevant choice of bound variable names when manipulating terms.