Composition of Unrestricted Macro Tree Transducers

Janis Voigtländer

$$
\begin{aligned}
&\text{data}\ \mathsf{Nat} \ = \ \mathsf{Z} \mid \mathsf{S}\ \mathsf{Nat} \\
&\text{data}\ \mathsf{List} \ = \ \mathsf{E} \mid \mathsf{A}\ \mathsf{List} \mid \mathsf{B}\ \mathsf{List} \\
\\
&ex :: \mathsf{List} \ \rightarrow \ \mathsf{List} \\
&ex\ (\mathsf{A}\ x) = \ \mathsf{B}\ (ex\ x) \\
&ex\ (\mathsf{B}\ x) = \ \mathsf{A}\ (ex\ x) \\
&ex \quad \mathsf{E} \quad = \mathsf{E} \\
\\
&rev :: \mathsf{List} \ \rightarrow \ \mathsf{List} \ \rightarrow \ \mathsf{List} \\
&rev\ (\mathsf{A}\ x)\ y \ = \ rev\ x\ (\mathsf{A}\ y) \\
&rev\ (\mathsf{B}\ x)\ y \ = \ rev\ x\ (\mathsf{B}\ y) \\
&rev \quad \mathsf{E} \quad y \ = \ y \\
\\
&exp :: \mathsf{Nat} \ \rightarrow \ \mathsf{Nat} \ \rightarrow \ \mathsf{Nat} \\
&exp\ (\mathsf{S}\ x)\ y \ = \ exp\ x\ (exp\ x\ y) \\
&exp \quad \mathsf{Z} \quad y \ = \ \mathsf{S}\ y
\end{aligned}
$$

Composition Techniques:

$ex\ (ex\ l)$	$TOP\ ;\ TOP \Rightarrow TOP$	[Rou70]
$rev\ (ex\ l)\ \mathsf{E}$	$TOP\ ;\ MAC \Rightarrow MAC$	[Eng81]
$ex\ (rev\ l\ \mathsf{E})$	$MAC\ ;\ TOP \Rightarrow MAC$	[EV85]
$rev\ (rev\ l\ \mathsf{E})\ \mathsf{E}$	$MAC_{nc}\ ;\ MAC_{wsu} \Rightarrow MAC$	[Küh98, Voi01]
$exp\ (exp\ n\ \mathsf{Z})\ \mathsf{Z}$	$MAC\ ;\ MAC \Rightarrow\ ???$	

$\boxed{TOP\ ;TOP \Rightarrow TOP}$

Translate right-hand sides of first transducer with rules of second one!

$$\boxed{\begin{array}{llll} p & (\mathsf{C}\ x_1 \cdots x_k) & \Rightarrow & rhs_{p,\mathsf{C}} \\ p & (q\ x) & \Rightarrow & \overline{qp}\ x \end{array}}$$

$$\begin{array}{lcl} \overline{ex\,ex}\ (\mathsf{A}\ x) & = & ex\ (\mathsf{B}\ (ex\ x)) \\ & \Rightarrow & \mathsf{A}\ (ex\ (ex\ x)) \\ & \Rightarrow & \mathsf{A}\ (\overline{ex\,ex}\ x) \\ \overline{ex\,ex}\ \mathsf{E} & = & ex\ \mathsf{E} \\ & \Rightarrow & \mathsf{E} \end{array}$$

Replace occurrences of $(ex\ (ex\ l))$ by $(\overline{ex\,ex}\ l)$.

$\boxed{TOP \; ; \; MAC \; \Rightarrow \; MAC}$

$$\boxed{\begin{array}{llll} p \; (\mathsf{C} \; x_1 \cdots x_k) & z_1 \cdots z_s & \Rightarrow & rhs_{p,\mathsf{C}} \\ p \quad\; (q \; x) & z_1 \cdots z_s & \Rightarrow & \overline{q\,p} \; x \; z_1 \cdots z_s \end{array}}$$

$$\begin{array}{lllll} \overline{ex\,rev} & (\mathsf{A} \; x) & z & = & rev \; (\mathsf{B} \; (ex \; x)) \; z \\ & & & \Rightarrow & rev \; (ex \; x) \; (\mathsf{B} \; z) \\ & & & \Rightarrow & \overline{ex\,rev} \; x \; (\mathsf{B} \; z) \\ \overline{ex\,rev} & \mathsf{E} & z & = & rev \; \mathsf{E} \; z \\ & & & \Rightarrow & z \end{array}$$

Replace occurrences of $(rev \; (ex \; l) \; \mathsf{E})$ by $(\overline{ex\,rev} \; l \; \mathsf{E})$.

$\boxed{MAC \; ; TOP \;\Rightarrow\; MAC}$

$$\boxed{\begin{array}{llll} p & (\mathsf{C}\; x_1 \cdots x_k) & \Rightarrow & rhs_{p,\mathsf{C}} \\ p & (q\; x\; y_1 \cdots y_r) & \Rightarrow & \overline{qp}\; x\; (p_1\; y_1) \;\cdots\; (p_m\; y_r) \\ p & \quad y_h & \Rightarrow & y_{h,p} \end{array}}$$

$$\begin{array}{llllll} \overline{rev\, ex} & (\mathsf{A}\; x) & y_{ex} & = & ex\; (rev\; x\; (\mathsf{A}\; y)) \\ & & & \Rightarrow & \overline{rev\, ex}\; x\; (ex\; (\mathsf{A}\; y)) \\ & & & \Rightarrow & \overline{rev\, ex}\; x\; (\mathsf{B}\; (ex\; y)) \\ & & & \Rightarrow & \overline{rev\, ex}\; x\; (\mathsf{B}\; y_{ex}) \\ \overline{rev\, ex} & \mathsf{E} & y_{ex} & = & ex\; y \\ & & & \Rightarrow & y_{ex} \end{array}$$

Replace occurrences of $(ex\; (rev\; l\; \mathsf{E}))$ by $(\overline{rev\, ex}\; l\; (ex\; \mathsf{E}))$.

MAC_{nc} ; MAC_{wsu} $\Rightarrow$ MAC (1)

$$
\begin{array}{llllll}
p & (\mathsf{C}\ x_1 \cdots x_k) & z_1 \cdots z_s & \Rightarrow & rhs_{p,\mathsf{C}} & \\
p & (q\ x\ y_1 \cdots y_r) & z_1 \cdots z_s & \Rightarrow & \overline{qp}\ x\ (p_1\ \ y_1\ (\overline{1_q 1_{p_1}}\ x\ \cdots) & \\
 & & & & & \cdots \\
 & & & & & (\overline{1_q s_{1\,p_1}}\ x\ \cdots)) \\
 & & & & \quad\cdots & \\
 & & & & (p_m\ y_r\ (\overline{r_q 1_{p_m}}\ x\ \cdots) & \\
 & & & & & \cdots \\
 & & & & & (\overline{r_q s_{m\,p_m}}\ x\ \cdots)) \\
 & & & & z_1 \cdots z_s & \\
p & y_h & z_1 \cdots z_s & \Rightarrow & y_{h,p} &
\end{array}
$$

Plus rather involved technique to construct the rules for $\overline{h_q l_p}$-states !

$\boxed{MAC_{nc}\ ;\ MAC_{wsu}\ \Rightarrow\ MAC\ (2)}$

$$
\begin{array}{llllll}
\overline{rev\,rev} & (\mathsf{A}\ x) & y_{rev}\ z & = & rev\ (rev\ x\ (\mathsf{A}\ y))\ z \\
 & & & \Rightarrow & \overline{rev\,rev}\ x\ (rev\ (\mathsf{A}\ y)\ (\overline{1_{rev}1_{rev}}\ x\ \mathsf{E}\ z))\ z \\
 & & & \Rightarrow & \overline{rev\,rev}\ x\ (rev\ y\ (\mathsf{A}\ (\overline{1_{rev}1_{rev}}\ x\ \mathsf{E}\ z)))\ z \\
 & & & \Rightarrow & \overline{rev\,rev}\ x\ y_{rev}\ z \\
\overline{rev\,rev} & \mathsf{E} & y_{rev}\ z & = & rev\ y\ z \\
 & & & \Rightarrow & y_{rev} \\
\overline{1_{rev}1_{rev}} & (\mathsf{A}\ x) & y_{rev}\ z_{rev} & = & \cdots \\
 & & & = & \mathsf{A}\ (\overline{1_{rev}1_{rev}}\ x\ \mathsf{E}\ z_{rev})
\end{array}
$$

Replace occurrences of $(rev\ (rev\ l\ \mathsf{E})\ \mathsf{E})$ by $(\overline{rev\,rev}\ l\ (\overline{1_{rev}1_{rev}}\ l\ \mathsf{E}\ \mathsf{E})\ \mathsf{E})$.

$\boxed{MAC \ ; MAC \Rightarrow ??? \ (1)}$

Do NOT try to determine the context parameters of p's before they are needed !

$$\boxed{\begin{array}{lllll} p & (\mathsf{C}\ x_1 \cdots x_k) & z_1 \cdots z_s & \Rightarrow & rhs_{p,\mathsf{C}} \\ p & (q\ x\ y_1 \cdots y_r) & z_1 \cdots z_s & \Rightarrow & \overline{q\,p}\ x\ (p_1\ y_1)\ \cdots\ (p_m\ y_r)\ z_1 \cdots z_s \\ p & y_h & z_1 \cdots z_s & \Rightarrow & y_{h,p}\ z_1 \cdots z_s \end{array}}$$

Partial applications give rise to higher-order functions (not a *MAC*) !

Can we still eliminate intermediate results in accumulating parameters ?

YES !

$MAC \; ; \; MAC \Rightarrow ???$ (2)

Read $\quad p \; (\mathsf{C}\ x_1 \cdots x_k)\ z_1 \cdots z_s \quad = \quad rhs_{p,\mathsf{C}}$

as: $\quad p \; (\mathsf{C}\ x_1 \cdots x_k) \quad \Rightarrow \quad (\lambda z_1 \cdots z_s \rightarrow rhs_{p,\mathsf{C}})$!

$$\boxed{\begin{array}{lcl} p \;\; (\mathsf{C}\ x_1 \cdots x_k) & \Rightarrow & (\lambda z_1 \cdots z_s \rightarrow rhs_{p,\mathsf{C}}) \\ p \; (q\ x\ y_1 \cdots y_r) & \Rightarrow & \overline{q\,p}\ x\ (p_1\ y_1)\ \cdots\ (p_m\ y_r) \\ p \qquad\quad y_h & \Rightarrow & y_{h,p} \end{array}}$$

Example: $exp\ (exp\ n\ \mathsf{Z})\ \mathsf{Z}$

$$
\begin{array}{lll}
\overline{exp\,exp}\ (\mathsf{S}\ x)\ y_{exp}\ z & = & exp\ (exp\ x\ (exp\ x\ y))\ z \\
 & \Rightarrow & \overline{exp\,exp}\ x\ (exp\ (exp\ x\ y))\ z \\
 & \Rightarrow & \overline{exp\,exp}\ x\ (\overline{exp\,exp}\ x\ (exp\ y))\ z \\
 & \Rightarrow & \overline{exp\,exp}\ x\ (\overline{exp\,exp}\ x\ y_{exp})\ z \\
\overline{exp\,exp}\ \ \mathsf{Z}\ \ y_{exp}\ z & = & exp\ (\mathsf{S}\ y)\ z \\
 & \Rightarrow & (\lambda z \to exp\ y\ (exp\ y\ z))\ z \\
 & \Rightarrow & (\lambda z \to y_{exp}\ (exp\ y\ z))\ z \\
 & \Rightarrow & (\lambda z \to y_{exp}\ (y_{exp}\ z))\ z \\
 & \Rightarrow_\beta & y_{exp}\ (y_{exp}\ z)
\end{array}
$$

Replace $(exp\ (exp\ n\ \mathsf{Z})\ \mathsf{Z})$ by $(\overline{exp\,exp}\ n\ (exp\ \mathsf{Z})\ \mathsf{Z})$.

No intermediate result is produced !

Further Example: $e = (rev\ (rev\ l\ \mathsf{E})\ \mathsf{E})$

Compare the transformation result:

$$
\begin{aligned}
&\overline{rev\,rev} :: \mathsf{List} \to (\mathsf{List} \to \mathsf{List}) \to \mathsf{List} \to \mathsf{List}\\
&\overline{rev\,rev}\ (\mathsf{A}\ x)\ y_{rev}\ z = \overline{rev\,rev}\ x\ (\lambda u \to y_{rev}\ (\mathsf{A}\ u))\ z\\
&\overline{rev\,rev}\ (\mathsf{B}\ x)\ y_{rev}\ z = \overline{rev\,rev}\ x\ (\lambda u \to y_{rev}\ (\mathsf{B}\ u))\ z\\
&\overline{rev\,rev}\ \ \mathsf{E}\ \ y_{rev}\ z = y_{rev}\ z\\
&e' = \overline{rev\,rev}\ l\ (\lambda u \to u)\ \mathsf{E}
\end{aligned}
$$

with the result of the construction from [Voi01]:

$$
\begin{aligned}
&\overline{rev\,rev} :: \mathsf{List} \to \mathsf{List} \to \mathsf{List} \to \mathsf{List}\\
&\overline{rev\,rev}\ (\mathsf{A}\ x)\ y_{rev}\ z = \overline{rev\,rev}\ x\ y_{rev}\ z\\
&\overline{rev\,rev}\ (\mathsf{B}\ x)\ y_{rev}\ z = \overline{rev\,rev}\ x\ y_{rev}\ z\\
&\overline{rev\,rev}\ \ \mathsf{E}\ \ y_{rev}\ z = y_{rev}\\
&\overline{1_{rev}1_{rev}} :: \mathsf{List} \to \mathsf{List} \to \mathsf{List} \to \mathsf{List}\\
&\overline{1_{rev}1_{rev}}\ (\mathsf{A}\ x)\ y_{rev}\ z_{rev} = \mathsf{A}\ (\overline{1_{rev}1_{rev}}\ x\ \mathsf{E}\ z_{rev})\\
&\overline{1_{rev}1_{rev}}\ (\mathsf{B}\ x)\ y_{rev}\ z_{rev} = \mathsf{B}\ (\overline{1_{rev}1_{rev}}\ x\ \mathsf{E}\ z_{rev})\\
&\overline{1_{rev}1_{rev}}\ \ \mathsf{E}\ \ y_{rev}\ z_{rev} = z_{rev}\\
&e' = \overline{rev\,rev}\ l\ (\overline{1_{rev}1_{rev}}\ l\ \mathsf{E}\ \mathsf{E})\ \mathsf{E}
\end{aligned}
$$

As a macro attributed tree transducer:

$$
\begin{array}{l}
\text{beginmag } \mathit{RevRev} \\
\quad \text{input } \mathit{Data} \\
\quad \text{root } \ \ \mathsf{Root} \\
\quad \text{syn } \ \ e' \qquad\qquad :: \ \mathsf{Root} \ \rightarrow \ \mathsf{List} \\
\quad \text{syn } \ \ \overline{\mathit{revrev}} \ :: \ \mathsf{List} \ \rightarrow \ \mathsf{List} \\
\quad \text{inh } \ \ y_{rev} \qquad :: \ \mathsf{List} \ \rightarrow \ \mathsf{List} \ \rightarrow \ \mathsf{List} \\
\quad \text{inh } \ \ z \qquad\qquad :: \ \mathsf{List} \ \rightarrow \ \mathsf{List} \\
\quad r@(\mathsf{Root}\ l) \text{ in} \\
\qquad e' \quad r \quad = \overline{\mathit{revrev}}\ l \\
\qquad y_{rev}\ l\ u = u \\
\qquad z \quad l \quad = \mathsf{E} \\
\quad l@(\mathsf{A}\ x) \text{ in} \\
\qquad \overline{\mathit{revrev}}\ l \quad = \overline{\mathit{revrev}}\ x \\
\qquad y_{rev} \quad x\ u = y_{rev}\ l\ (\mathsf{A}\ u) \\
\qquad z \qquad x \quad = z\ l \\
\quad l@(\mathsf{B}\ x) \text{ in} \\
\qquad \overline{\mathit{revrev}}\ l \quad = \overline{\mathit{revrev}}\ x \\
\qquad y_{rev} \quad x\ u = y_{rev}\ l\ (\mathsf{B}\ u) \\
\qquad z \qquad x \quad = z\ l \\
\quad l@\mathsf{E} \text{ in} \\
\qquad \overline{\mathit{revrev}}\ l = y_{rev}\ l\ (z\ l) \\
\text{endmag}
\end{array}
$$

References

[Eng81] J. Engelfriet. Tree transducers and syntax directed semantics. Technical Report Memorandum 363, Technische Hogeschool Twente, 1981.

[EV85] J. Engelfriet and H. Vogler. Macro tree transducers. *J. Comput. Syst. Sci.*, 31:71–145, 1985.

[Küh98] A. Kühnemann. Benefits of tree transducers for optimizing functional programs. In *Foundations of Software Technology & Theoretical Computer Science, Chennai, India, Proceedings*, volume 1530 of *LNCS*, pages 146–157. Springer-Verlag, December 1998.

[Rou70] W.C. Rounds. Mappings and grammars on trees. *Math. Syst. Theory*, 4:257–287, 1970.

[Voi01] J. Voigtländer. Composition of restricted macro tree transducers. Master's thesis, Dresden University of Technology, March 2001. *http://wwwtcs.inf.tu-dresden.de/∼voigt/mscthesis.ps.gz* (revised version).