Übersetzung in case-Ausdrücke

Gegeben: Funktionsdefinition

$$
\begin{array}{l}
f\ \ p_{11} \ldots p_{1n}\ =\ e_1 \\
\vdots \\
f\ \ p_{m1} \ldots p_{mn}\ =\ e_m
\end{array}
$$

Übersetzung (x_i neu):

$$
\begin{array}{l}
\texttt{f}\ \ x_1 \ldots x_n\ = \\
\quad [\![\texttt{match}\ \ [x_1,\ldots,x_n]\ \ [([p_{11},\ldots,p_{1n}],\ e_1), \\
\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\qquad\qquad\qquad\qquad\qquad\qquad ([p_{m1},\ldots,p_{mn}],\ e_m)] \\
\qquad\qquad\qquad\qquad\qquad\quad \texttt{ERROR}]\!]
\end{array}
$$

Transformationsregel 1

Alle Muster fangen mit einer Variablen an:

$$
\begin{aligned}
&[\![\texttt{match (x:xs) [(}v_1 : ps_1,\ e_1\texttt{),}\\
&\qquad\qquad\qquad\vdots\\
&\qquad\qquad\qquad\texttt{(}v_m : ps_m,\ e_m\texttt{)] E}]\!]\\
&= [\![\texttt{match xs [(}ps_1,\ e_1[v_1 \mapsto \texttt{x}]\texttt{),}\\
&\qquad\qquad\qquad\vdots\\
&\qquad\qquad\qquad\texttt{(}ps_m,\ e_m[v_m \mapsto \texttt{x}]\texttt{)] E}]\!]
\end{aligned}
$$

$e[v \mapsto t]$: Ersetzung aller Vorkommen von v in e durch t

$m \geq 0$: auch anwendbar für $m = 0$!

Transformationsregel 2

Alle Muster fangen mit einem Konstruktor an:

$[\![\texttt{match (x:xs) eqs E}]\!] = \texttt{-- eqs} \neq \texttt{[]}$

Idee: gruppiere $\texttt{eqs} = (\texttt{eqs}_1 \texttt{ ++ } \cdots \texttt{ ++ } \texttt{eqs}_k)$ mit

$$\texttt{eqs}_i = [((C_i \;\; p_{i,1,1} \ldots p_{i,1,n_i}) : ps_{i,1}, \;\; e_{i,1}),$$
$$\vdots$$
$$((C_i \;\; p_{i,m_i,1} \ldots p_{i,m_i,n_i}) : ps_{i,m_i}, \;\; e_{i,m_i})]$$

$C_1, \ldots, C_k$ alle Konstruktoren vom Typ von $\texttt{x}$

$\texttt{eqs}_j = []$ falls in eqs der Konstruktor C_j nicht vorkommt

Transformationsregel 2

Alle Muster fangen mit einem Konstruktor an:

$$
\begin{array}{l}
[\![\texttt{match (x:xs) eqs E}]\!] = \texttt{ -- eqs}\neq\texttt{[], x}_{i,j}\texttt{ neu} \\
\quad \texttt{case x of} \\
\qquad C_1\ x_{1,1} \ldots x_{1,n_1} \rightarrow [\![\texttt{match ([}x_{1,1},\ldots,x_{1,n_1}\texttt{]++xs) }eqs'_1\texttt{ E}]\!] \\
\qquad \vdots \\
\qquad C_k\ x_{k,1} \ldots x_{k,n_k} \rightarrow [\![\texttt{match ([}x_{k,1},\ldots,x_{k,n_k}\texttt{]++xs) }eqs'_k\texttt{ E}]\!]
\end{array}
$$

wobei

$$
\begin{array}{rcl}
eqs'_i & = & [([p_{i,1,1},\ldots,p_{i,1,n_i}] \texttt{ ++ } ps_{i,1}\texttt{, } e_{i,1})\texttt{,} \\
& & \qquad \vdots \\
& & \quad ([p_{i,m_i,1},\ldots,p_{i,m_i,n_i}] \texttt{ ++ } ps_{i,m_i}\texttt{, } e_{i,m_i})]
\end{array}
$$

Transformationsregel 3

Argument-/Musterlisten sind leer:

1. Genau eine Regel anwendbar:

   $$[\![\texttt{match [] [([], e)] E}]\!] = \texttt{e}$$

2. Keine Regel anwendbar: Benutze Fehleralternative:

   $$[\![\texttt{match [] [] E}]\!] = [\![\texttt{E}]\!]$$

3. Mehr als eine Regel anwendbar:
   Fehler, Warnung oder erste Regel anwenden

Transformationsregel 4

Musterlisten mit Konstruktoren und auch Variablen:

Gruppiere Regeln nach Konstruktoren und Variablen (siehe unten):

$$[\![\texttt{match xs eqs E}]\!] =$$
$$[\![\texttt{match xs eqs}_1\ \texttt{(match xs eqs}_2\ \texttt{(...(match xs eqs}_n\ \texttt{E) ...))}]\!]$$

mit $\texttt{eqs} == \texttt{eqs}_1 \texttt{ ++ eqs}_2 \texttt{ ++ ...++ eqs}_n$ und
$\forall i \in \{1, \ldots, n\} : \texttt{eqs}_i \neq []$ und:

- Entweder beginnen die Muster in allen Regeln in $\texttt{eqs}_i$ mit einer Variablen und, falls $i < n$, in allen Regeln in $\texttt{eqs}_{i+1}$ beginnen die Muster mit Konstruktoren,
- oder die Muster in allen Regeln in $\texttt{eqs}_i$ beginnen mit einem Konstruktor und, falls $i < n$, in allen Regeln in $\texttt{eqs}_{i+1}$ beginnen die Muster mit Variablen.