Delta-Rules

A set of rules δ of the form $c\,t_1 \ldots t_n \to r$
with $c \in \mathcal{C},\ t_1, \ldots, t_n,\ r \in \Lambda$ is a *set of Delta-rules* iff

- $t_1, \ldots, t_n, r$ are closed lambda-terms
- all t_i are in $\to_\beta$-normal form and
  they do not contain the left-hand side of a rule from δ
- δ does not contain rules $c\,t_1 \ldots t_n \to r$ and $c\,t_1 \ldots t_m \to r'$
  with $m \geq n$ and $r \neq r'$

δ-Reduction

- $l \to_\delta r$, if $l \to r \in \delta$
- $t_1 \to_\delta t_2$ implies $(t_1\,r) \to_\delta (t_2\,r)$, $(r\,t_1) \to_\delta (r\,t_2)$, $\lambda y.t_1 \to_\delta \lambda y.t_2$

We define $\to_{\beta\delta} = \to_\beta \cup \to_\delta$.

Non-Terminating Reductions

- Even β-reduction does not terminate:

$$(\lambda x.x\,x)\,(\lambda x.x\,x) \;\rightarrow_\beta\; (\lambda x.x\,x)\,(\lambda x.x\,x) \;\rightarrow_\beta\; \ldots$$

- In general, termination depends on the reduction strategy.
  - Leftmost outermost reduction:

$$(\lambda x.y)((\lambda x.x\,x)\,(\lambda x.x\,x)) \;\rightarrow_\beta\; y,$$

  - Leftmost innermost reduction:

$$(\lambda x.y)((\lambda x.x\,x)\,(\lambda x.x\,x)) \;\rightarrow_\beta\; (\lambda x.y)((\lambda x.x\,x)\,(\lambda x.x\,x)) \;\rightarrow_\beta\; \ldots$$

Translation of Simple HASKELL into Lambda Terms

$\mathcal{C}_0 =$ pre-defined function symbols (`+`, `not`, `sqrt`, etc.)

Con $=$ constructor symbols (including `Int`, `Float`, `Char`)

$\mathcal{C} = \mathcal{C}_0 \cup \text{Con} \cup \{\texttt{tuple}_n \mid n \in \{0, 2, 3, \ldots\}\} \cup \{\texttt{if}, \texttt{fix}\}$

$$
\begin{array}{lcl}
\mathcal{L}am(\underline{\text{var}}) & = & \underline{\text{var}} \\
\mathcal{L}am(c) & = & c, \quad \text{where } c \in \mathcal{C}_0 \cup \text{Con} \\
\mathcal{L}am((\underline{\text{exp}}_1, \ldots, \underline{\text{exp}}_n)) & = & \texttt{tuple}_n\, \mathcal{L}am(\underline{\text{exp}}_1)\ \ldots\ \mathcal{L}am(\underline{\text{exp}}_n), \\
 & & \qquad \text{where } n \in \{0, 2, 3, \ldots\} \\
\mathcal{L}am((\underline{\text{exp}})) & = & \mathcal{L}am(\underline{\text{exp}}) \\
\mathcal{L}am((\underline{\text{exp}}_1\ \underline{\text{exp}}_2)) & = & (\mathcal{L}am(\underline{\text{exp}}_1)\, \mathcal{L}am(\underline{\text{exp}}_2)) \\
\mathcal{L}am(\texttt{if}\ \underline{\text{exp}}_1\ \texttt{then}\ \underline{\text{exp}}_2\ \texttt{else}\ \underline{\text{exp}}_3) & = & \texttt{if}\, \mathcal{L}am(\underline{\text{exp}}_1)\, \mathcal{L}am(\underline{\text{exp}}_2)\, \mathcal{L}am(\underline{\text{exp}}_3) \\
\mathcal{L}am(\texttt{let}\ \underline{\text{var}} = \underline{\text{exp}}\ \texttt{in}\ \underline{\text{exp}}') & = & \mathcal{L}am(\underline{\text{exp}}')\ [\underline{\text{var}}\,/\,(\texttt{fix}\,(\lambda \underline{\text{var}}.\mathcal{L}am(\underline{\text{exp}})))] \\
\mathcal{L}am(\backslash \underline{\text{var}}\ \texttt{->}\ \underline{\text{exp}}) & = & \lambda \underline{\text{var}}.\mathcal{L}am(\underline{\text{exp}})
\end{array}
$$

δ-Rules for HASKELL-Programs

Con_n = constructor symbols of arity n

δ_0 = rules for pre-defined symbols in HASKELL

$$
\begin{aligned}
\delta \;=\; & \delta_0 \cup \\
& \{\mathtt{bot} \to \mathtt{bot}, \\
& \mathtt{isa}_{()}\,\mathtt{tuple}_0 \to \mathtt{True}, \\
& \mathtt{if}\,\mathtt{True} \to \lambda xy.x, \\
& \mathtt{if}\,\mathtt{False} \to \lambda xy.y, \\
& \mathtt{fix} \to \lambda f.\, f(\mathtt{fix}\, f)\} \cup \\
& \{\mathtt{isa}_{\underline{\mathsf{constr}}}\,(\underline{\mathsf{constr}}\, t_1 \ldots t_n) \to \mathtt{True} \mid \underline{\mathsf{constr}} \in \mathsf{Con}_n,\, t_j \text{ closed}\} \cup \\
& \{\mathtt{isa}_{\underline{\mathsf{constr}}}\,(\underline{\mathsf{constr}'}\, t_1 \ldots t_m) \to \mathtt{False} \mid \underline{\mathsf{constr}} \neq \underline{\mathsf{constr}'} \in \mathsf{Con}_m,\, t_j \text{ closed}\} \cup \\
& \{\mathtt{argof}_{\underline{\mathsf{constr}}}\,(\underline{\mathsf{constr}}\, t_1 \ldots t_n) \to \mathtt{tuple}_n\, t_1 \ldots t_n \mid \underline{\mathsf{constr}} \in \mathsf{Con}_n, \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad n \in \{0, 2, 3, \ldots\}, t_j \text{ closed}\} \cup \\
& \{\mathtt{argof}_{\underline{\mathsf{constr}}}\,(\underline{\mathsf{constr}}\, t) \to t \mid \underline{\mathsf{constr}} \in \mathsf{Con}_1,\, t \text{ closed}\} \cup \\
& \{\mathtt{sel}_{n,i}\,(\mathtt{tuple}_n\, t_1 \ldots t_n) \to t_i \mid n \geq 2,\, 1 \leq i \leq n,\, t_j \text{ closed}\}
\end{aligned}
$$