

Prof. Dr. Jürgen Giesl
René Thiemann

Exercises *Functional Programming* – Sheet 12

Solutions will be collected until Thursday, Jan 23, 2003 in the exercise course.

Exercise 1 (0.25 + 0.25 + 1 + 0.5 points)

The notion of “ δ -reduction” is defined as follows:

A set δ of rules of the form $ct_1 \dots t_n \rightarrow r$ with $c \in \mathcal{C}$, $t_1, \dots, t_n, r \in \Lambda$ is called a delta-rule set if

- (1) t_1, \dots, t_n, r are closed lambda terms
- (2) all t_i are in \rightarrow_β -normal form
- (3) the t_i do not contain any left-hand side of a rule from δ
- (4) in δ there exist no two rules $ct_1 \dots t_n \rightarrow r$ and $ct_1 \dots t_m \rightarrow r'$ with $m \geq n$

For such a set δ we define the relation \rightarrow_δ as the least relation with

- $l \rightarrow_\delta r$ for all $l \rightarrow r \in \delta$
- if $t_1 \rightarrow_\delta t_2$, then also $(t_1 r) \rightarrow_\delta (t_2 r)$, $(r t_1) \rightarrow_\delta (r t_2)$ and $\lambda y.t_1 \rightarrow_\delta \lambda y.t_2$ for all $r \in \Lambda$, $y \in \mathcal{V}$.

We denote the combination of β - and δ -reduction by $\rightarrow_{\beta\delta}$, i.e., $\rightarrow_{\beta\delta} = \rightarrow_\beta \cup \rightarrow_\delta$.

Please find counterexamples to show that all four conditions (1), (2), (3), (4) are necessary for the uniqueness of normal forms of the $\rightarrow_{\beta\delta}$ -relation. This means if one condition is not valid and the other three hold, uniqueness of normal forms is not guaranteed.

Exercise 2 (0.5 + 0.5 + 1 points)

For each of the following terms please write down the weak head normal form and show the reduction steps of the WHNO-reduction with the $\rightarrow_{\beta\delta}$ -relation, where δ is the set of rules from Definition 3.3.5:

- (a) $(\lambda x y.(\lambda z.z) y (\text{plus } x 1)) 5$
- (b) $(\lambda x y.x (\lambda z.y z)) ((\lambda x y.y) 8 (\lambda x.(\lambda y.y) x))$
- (c) $(\lambda h.(\lambda x.h (x x)) (\lambda x.h (x x))) ((\lambda x.x) (\text{plus } 1 5))$

Exercise 3 (1 + 1.5 + 1 + 1 + 1.5 + 1 + 0.5 + 0.5 points)

Please translate the following Haskell-expressions into lambda terms using \mathcal{Lam} and reduce them to weak head normal form with the $\rightarrow_{\beta\delta}$ -relation, where δ is the set of rules from Definition 3.3.5:

- (a) `let not' = \x -> if x == True then False else True in not' True`
- (b) `let double = \x -> if x == 0 then 0 else (plus 2 (double (x - 1)))
in double 1`
- (c) `let double = \x -> if isaZero x then Zero
else Succ (Succ (double (argofSucc x)))
in double (Succ (Succ (Zero)))`
- (d) `let double = \x -> if isaZero x then Zero
else Succ (Succ (double (argofSucc x)))
in double (Succ (Succ (bot)))`
- (e) `let length = \xs -> if isaNil xs then 0
else 1 + (length (sel2,2 (argofCons xs)))
in length (Cons bot Nil)`
- (f) `let f = \x -> x + x in f (f 3)`
- (g) `let inf = 1 + inf in inf`
- (h) `let inf = Succ inf in inf`

Hint: You can write `(minus x 1)` instead of `(x - 1)`, `(eq x 0)` instead of `x == 0`, etc.

Exercise 4 (1 + 1 + 1.5 + 2.5 points)

Please implement the following missing parts in the given framework for WHNO-reduction of $\rightarrow_{\beta\delta}$.

- \mathcal{Lam}
- The WHNO-Strategy
- Parts of δ_0 : not, binary integer functions
- Parts of $\delta \setminus \delta_0$: fix, isa_{xxx}, argof_{xxx}

Please turn around

In the implementation part, we refine beta-reduction to „ $(\lambda x.e e') \rightarrow_{\beta} e[x/e']$ “ and define $\rightarrow'_{\beta\delta}$ as $\rightarrow_{\beta} \cup \delta$. So, these reductions are only applicable on the top-most level of a term. In order to apply $\rightarrow'_{\beta\delta}$ inside the terms, we need a strategy that says us, where to apply $\rightarrow'_{\beta\delta}$. These strategies can be leftmost-outermost, leftmost-innermost, everywhere, ... It holds, that $strat_{io}(\rightarrow'_{\beta\delta}) = \rightarrow_{\beta\delta}$, where $\rightarrow_{\beta\delta}$ is the relation from the lecture. So, it is a valid implementation if we first implement only the top-most-reductions and afterwards apply some strategy on them.

The framework can be downloaded from the website. You only have to modify the file "Delta.hs" at the marked positions, but you should look in the headers of the other files to see, what functions are available. All examples of this exercise sheet are available for testing at the end of "Delta.hs".

Remark:

Definition 3.3.5 (δ -rules for HASKELL-programs) Let Con be the set of constructors of a HASKELL-program, where Con_n denotes the set of all constructors of arity n . Let δ_0 be the rules of the predefined operators in HASKELL, i.e., δ_0 contains rules such as $\text{plus } 1 \ 5 \rightarrow 6$, $\text{not True} \rightarrow \text{False}$, etc. All these rules have the form $f t_1 \dots t_n \rightarrow r$, where $f \in \mathcal{C}_0$ and $t_1, \dots, t_n, r \in \text{Con}_0$. Because there are not two rules with the same left-hand side, δ_0 is a delta-rule set. The entire set δ is defined as follows:

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