

Prof. Dr. Jürgen Giesl  
Peter Schneider-Kamp

## Exercises *Functional Programming* – Sheet 12

Solutions will be collected until Wednesday, July 20, 2005 in the exercise course.

Exercises can be solved both in English and in German.

### Exercise 1 (4 points)

In the translations from Haskell to simple Haskell, we used twelve transformation rules. Rule (5) was given as:

$$\frac{\text{match } \underline{\text{var}} \ \underline{\text{exp}} \ \underline{\text{exp}_1} \ \underline{\text{exp}_2}}{(\underline{\text{var}} \rightarrow \underline{\text{exp}_1}) \ \underline{\text{exp}}}$$

We now replace the transformation rule (5) by the following one:

$$\frac{\text{match } \underline{\text{var}} \ \underline{\text{exp}} \ \underline{\text{exp}_1} \ \underline{\text{exp}_2}}{\underline{\text{exp}_1} \ [\underline{\text{var}}/\underline{\text{exp}}]}$$

Show that there exists a Haskell-expression such that this change leads to a well-typed lambda term, whereas the original transformation rules produce a non well-typed term.

### Important:

The written test (Übungsscheinklausur) will take place on Friday, July 22, 10am in AH II instead of the last lecture.

To get a certificate for this course (Übungsschein) you must pass this test *and* reach at least 50% of the points on the exercise sheets. We recommend the acquisition of this certificate, since this is a good opportunity to prepare for the diploma or master examination.

In the next two exercises, please use the initial type assumption  $A_0$  as presented in the lecture. This type assumption contains at least the following types:

$A_0(1)$	=	<code>Int</code>
$A_0(\text{True})$	=	<code>Bool</code>
$A_0(\text{False})$	=	<code>Bool</code>
$A_0(\text{plus})$	=	<code>Int → Int → Int</code>
$A_0(\text{Nil})$	=	$\forall a. \text{List } a$
$A_0(\text{Cons})$	=	$\forall a. a \rightarrow \text{List } a \rightarrow \text{List } a$
$A_0(\text{tuple}_2)$	=	$\forall a, b. a \rightarrow b \rightarrow (a, b)$
$A_0(\text{fix})$	=	$\forall a. (a \rightarrow a) \rightarrow a$
$A_0(\text{isa}_{\text{True}})$	=	<code>Bool → Bool</code>
$A_0(\text{if})$	=	$\forall a. \text{Bool} \rightarrow a \rightarrow a \rightarrow a$

## Exercise 2 (6 points)

Give the most general shallow type schema for each term if it exists. For the remaining terms you should say why they cannot be typed using a shallow type schema.

- (a)  $\lambda x y. \text{Cons } x y$
- (b)  $\lambda x y. \text{Cons Nil } (\text{Cons } x y)$
- (c)  $\lambda x. \text{Cons Nil } (\text{Cons } 1 x)$
- (d)  $\lambda x y. \text{plus } (x y) (y x)$

## Exercise 3 (3 + 3 + 4 points)

Use the type inference algorithm  $\mathcal{W}$  to determine the most general type of the following  $\lambda$ -terms. Show the results of all sub computations and unifications, too. If the term is not well-typed, show at what step and why the  $\mathcal{W}$ -algorithm detects this.

- (a)  $\lambda x. \text{tuple}_2 (x \text{ True}) (x 1)$
- (b)  $\text{plus } (x \text{ True}) (x 1)$
- (c)  $\text{fix } (\lambda n x. \text{if } (\text{isa}_{\text{True}} x) \text{ False True})$