Assignment 3
Polymorphism

15-814: Types and Programming Languages
Frank Pfenning

Due Tuesday September 29, 2020

This assignment is due on the above date and it must be submitted electronically as a PDF file on Canvas. You may use the distributed lecture notes and assignment sources to typeset your assignment and make sure to include your full name and Andrew ID on the written part.

You should hand in two files

- `hw03.pdf` with your written solutions to the questions
- `hw03.poly` with the code, where the solutions to the problems are clearly marked and auxiliary code (either from lecture or your own) is included so it passes the LAMBDA checker.

1 Polymorphic Functions

Task 1 (L6.1, 10 pts)

(i) Find a definition of *plus* : *nat* $\to$ *nat* $\to$ *nat* that works in the simply-typed λ-calculus in the sense that we need to instantiate the type $\forall\alpha.\,(\alpha \to \alpha) \to \alpha \to \alpha$ only with a type variable.

(ii) Give a simply-typed definition (in the sense of part (i)) for *times* or conjecture that none exists.

Include your definition(s) and a few test cases in the file `hw03.poly`.

Task 2 (L6.3, 15 pts) We write F for a (mathematical) function from types to types (which is not expressible in the polymorphic λ-calculus but requires system F^ω). A more general family of types (one for each F) for self-application is given by

$$\begin{array}{lcl} w_F & = & \forall\alpha.\,\alpha \to F(\alpha) \\ \omega_F & : & w_F \to F(w_F) \\ \omega_F & = & \lambda x.\,x\,[w_F]\,x \end{array}$$

We recover the type from this lecture with $F = \Lambda\alpha.\,\alpha$. You may want to verify the general typing derivation in preparation for the following questions, but you do not need to show it.

(i) Consider $F = \Lambda\alpha.\,\alpha \to \alpha$. In this case $w_F = \mathit{bool}$. Calculate the type and characterize the behavior of ω_F as a function on Booleans.

(ii) Consider $F = \Lambda\alpha.\,(\alpha \to \alpha) \to \alpha$. Calculate w_F, the type of ω_F, and characterize the the behavior of ω_F. Can you relate w_F and ω_F to the types and functions we have considered in the course so far?

For both parts, include your functions with their type in the file `hw03.poly` together with a few test cases that demonstrate your interpretation of how ω_F behaves in those two instances.

Task 3 (L7.1, part (i), 10 pts) Show the *new cases* in the proof of preservation and progress arising from parametric polymorphism.

(i) (Preservation) If $\Gamma \vdash e : \tau$ and $e \longrightarrow e'$ then $\Gamma \vdash e' : \tau$

(ii)-(iii) **not assigned**

Explicitly state any additional substitution properties you need (in addition to Theorem L5.6), but you do not need to prove them.

Task 4 (L7.2, 15 pts) An alternative form of binary tree to the one given in Lecture 7.3 is one where the natural numbers are stored in the leaves and not in the nodes. Let's call such a tree a *shrub*.

(i) Give the types for shrub constructors.

(ii) Give the construction of a shrub containing the numbers 1, 2, and 3.

(iii) Give the polymorphic definition of the type *shrub*, assuming it is represented by its own iterator.

(iv) Provide the definitions of the shrub constructors.

(v) Write a function *sumup* to sum the elements of a shrub.

(vi) Write a function *mirror* that returns the mirror image of a given tree, reflected about a vertical line down from the root.

Include your definitions in the file `hw03.poly` together with a few test cases.

Task 5 (L7.3 10 pts) We say two types τ and σ are *isomorphic* (written $\tau \cong \sigma$) if there are two functions *forth* $: \tau \to \sigma$ and *back* $: \sigma \to \tau$ such that they compose to the identity in both directions, that is, $\lambda x.\, \mathit{back}\ (\mathit{forth}\ x))$ is equal to $\lambda x.\, x$ and $\lambda y.\, \mathit{forth}\ (\mathit{back}\ y)$ is equal to $\lambda y.\, y$.

Consider the two types

$$\begin{array}{rcl} \mathit{nat} & = & \forall\alpha.\,(\alpha \to \alpha) \to \alpha \to \alpha \\ \mathit{tan} & = & \forall\alpha.\,\alpha \to (\alpha \to \alpha) \to \alpha \end{array}$$

(i) Provide functions *forth* $: \mathit{nat} \to \mathit{tan}$ and *back* $: \mathit{tan} \to \mathit{nat}$ that, intuitively, should witness the isomorphism between *nat* and *tan*.

(ii) Compute the normal forms of the two function compositions. You may recruit the help of the LAMBDA implementation for this purpose.

(iii) Are the two function compositions β-equal to the identity? If yes, you are done. If not, can you see a sense under which they would be considered equal, either by changing your two functions or be defining a suitably justified notion of equality?

Include your functions *forth* and *back* as well as their compositions in the file `hw03.poly`.