Práctica 4 - Sistemas de Tipos

Lenguaje de expresiones aritméticas

1. Demostrar mediante un árbol de derivación de tipado si los siguientes términos están bien tipados o no.

a) `succ (if iszero (succ 0) then 0 else succ 0)`

b) `if (if iszero 0 then false else true) then 0 else succ 0`

c) `if true then iszero 0 else succ 0`

d) `if false then (if succ 0 then true else false) else false`

2. Probar que el lenguaje de expresiones aritméticas visto en la teoría es seguro (mostrar progreso y preservación).

3. La propiedad de preservación nos dice que un término bien tipado evoluciona a otro término bien tipado. ¿Es verdad también la inversa? Probar que si $t \to t'$ y $t' \colon T$, entonces $t \colon T$, o dar un contraejemplo.

Lambda cálculo simple tipado

4. Demostrar mediante un árbol de derivación de tipado si los siguientes términos están bien tipados o no en el lambda cálculo simplemente tipado con booleanos.

a) $f \colon \texttt{Bool} \to \texttt{Bool} \vdash f\ (\texttt{if false then true else false}) \colon \texttt{Bool}$

b) $f \colon \texttt{Bool} \to \texttt{Bool} \vdash \lambda x\ .\ f\ (f\ \texttt{true}) \colon \texttt{Bool}$

c) $f \colon \texttt{Bool} \to \texttt{Bool} \vdash \lambda x\ .\ f\ (\texttt{if}\ x\ \texttt{then false else}\ x) \colon \texttt{Bool} \to \texttt{Bool}$

d) $f \colon \texttt{Bool} \to \texttt{Bool} \vdash \lambda g\ .\ \lambda x\ .\ g\ (f\ x) \colon (\texttt{Bool} \to \texttt{Bool}) \to \texttt{Bool} \to \texttt{Bool}$

e) $\mathit{and} \colon \texttt{Bool} \to \texttt{Bool} \to \texttt{Bool}, z \colon \texttt{Bool} \vdash \lambda x\ .\ \lambda y\ .\ \mathit{and}\ x\ (\mathit{and}\ z\ y) \colon \texttt{Bool}$

5. ¿Existe algún contexto Γ y tipo T tal que $\Gamma \vdash x\ x \colon T$? En caso afirmativo, dar Γ y T junto con la derivación de tipado que lo demuestre; en caso contrario, probarlo.

6. Considere el ćalculo lambda simple tipado con un único tipo base `Unit`, y una constante $\star \colon \texttt{Unit}$. ¿Es verdad que si $t \to t'$ y $t' \colon T$, entonces $t \colon T$? Justificar.

7. Estudiar la relación entre el λ-cálculo simple tipado a la Curry y a la Church.

a) Definir el λ-cálculo simple tipado a la Church.

b) Definir una función $|\_|$ que mapee un término a la Church en un término a la Curry mediante la eliminación de tipos en los términos.

c) Mostrar la correspondencia entre derivaciones de juicios de tipado a la Church y a la Curry, es decir, mostrar que:

- Si $\Gamma \vdash t' : T$ en el sistema a la Church, entonces $\Gamma \vdash |t'| : T$ en el sistema a la Curry.
- Si $\Gamma \vdash t : T$ en el sistema a la Curry, entonces existe t' tal que $t = |t'|$ y $\Gamma \vdash t' : T$ en el sistema a la Church.

Sistema T

8. Definir en T:

a) $\texttt{pred}\colon \texttt{Nat} \to \texttt{Nat}$: predecesor,

b) $\texttt{suma}\colon \texttt{Nat} \to \texttt{Nat} \to \texttt{Nat}$: suma de naturales,

c) $\texttt{mult}\colon \texttt{Nat} \to \texttt{Nat} \to \texttt{Nat}$: multiplicación de naturales,

d) $\texttt{is0}\colon \texttt{Nat} \to \texttt{Bool}$: comparación con $\texttt{0}$,

e) $\texttt{eq}\colon \texttt{Nat} \to \texttt{Nat} \to \texttt{Bool}$: igualdad de naturales.

9. Mostrar la relación entre T y las funciones recursivas primitivas:

a) Mostrar que toda función recursiva primitiva es definible en T.

b) Mostrar que existen funciones en T que no son primitivas recursivas.

Sistema F

10. Dadas las siguientes definiciones:

$$\begin{array}{l}\texttt{double}\colon \forall X\,.\ (X \to X) \to X \to X \\ \texttt{double} = \Lambda\, X\,.\lambda f\colon X \to X\,.\lambda x\colon X\,.\ f\ (f\ x)\end{array}$$

$$\begin{array}{l}\texttt{doubleNat}\colon (\texttt{Nat} \to \texttt{Nat}) \to \texttt{Nat} \to \texttt{Nat} \\ \texttt{doubleNat} = \texttt{double}\langle\texttt{Nat}\rangle\end{array}$$

$$\begin{array}{l}\texttt{doubleFun}\colon ((\texttt{Nat} \to \texttt{Nat}) \to \texttt{Nat} \to \texttt{Nat}) \to (\texttt{Nat} \to \texttt{Nat}) \to \texttt{Nat} \to \texttt{Nat} \\ \texttt{doubleFun} = \texttt{double}\langle\texttt{Nat} \to \texttt{Nat}\rangle\end{array}$$

$$\begin{array}{l}\texttt{id}\colon \forall X\,.\ X \to X \\ \texttt{id} = \Lambda\, X\,.\lambda x\colon X\,.\ x\end{array}$$

a) Probar que los términos están bien tipados

b) Usando `double` definir una función `quadruple` que aplique una función argumento cuatro veces.

11. Definir la conjunción de booleanos representados como $\texttt{Bool} = \forall X\,.\ X \to X \to X$.

12. Mostrar que el tipo $\texttt{PairNat} = \forall X\,.\ (\texttt{Nat} \to \texttt{Nat} \to X) \to X$ puede ser usado para representar pares de números. Para esto escribir funciones:

a) $\texttt{pairNat}\colon \texttt{Nat} \to \texttt{Nat} \to \texttt{PairNat}$

b) $\texttt{fstNat}\colon \texttt{PairNat} \to \texttt{Nat}$

c) $\texttt{sndNat}\colon \texttt{PairNat} \to \texttt{Nat}$

13. Definir la función predecesor de los naturals representados como $\texttt{Nat} = \forall X\,.\ (X \to X) \to X \to X$. Ayuda: Usar el tipo `PairNat` para implementar *tupling*.

14. Dada la representación de listas: $\texttt{List}\ X = \forall Y\,.\ (X \to Y \to Y) \to Y \to Y$

a) Definir y dar el tipo de la función `map` para listas.

b) Definir una función $\texttt{insert}\colon \forall X\,.\ (X \to X \to \texttt{Bool}) \to \texttt{List}\ X \to X \to \texttt{List}\ X$ que dada una función de comparación y una lista ordenada, inserta un elemento.

Probar las soluciones propuestas implementándolas en Haskell. Para esto habilitar la extensión `RankNTypes` poniendo al principio del archivo:

```
{-# LANGUAGE RankNTypes #-}
```