Práctico 7: Teoría de tipos de Martin-Löf

Ejercicio 1 (Las operaciones aritméticas y sus propiedades)**.**

(1) Construir a partir del recursor

$$\texttt{nat\_rec} \;:\; \forall X : \texttt{Nat} \to \mathsf{Set}.\; X\,\texttt{0} \to (\forall y : \texttt{Nat}.\, X\,y \to X\,(\texttt{S}\,y)) \to \forall x : \texttt{Nat}.\, X\,x$$

términos $\texttt{plus}, \texttt{mult} : \texttt{Nat} \to \texttt{Nat} \to \texttt{Nat}$ que cumplen las conversiones

$$\begin{array}{rcl@{\qquad}rcl} \texttt{plus}\,\texttt{0}\,m & \cong & m & \texttt{mult}\,\texttt{0}\,m & \cong & \texttt{0} \\ \texttt{plus}\,(\texttt{S}\,n)\,m & \cong & \texttt{S}\,(\texttt{plus}\,n\,m) & \texttt{mult}\,(\texttt{S}\,n)\,m & \cong & \texttt{plus}\,m\,(\texttt{mult}\,n\,m) \end{array}$$

para todos $n, m : \texttt{Nat}$. (¡Cuidado! Aquí se definen `plus` y `mult` por recursión sobre su primer argumento, al contrario de las diapositivas del curso.)

En lo siguiente, se escriben $n + m :\equiv \texttt{plus}\,n\,m$ y $n \times m :\equiv \texttt{mult}\,n\,m$.

(2) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
   a) $\forall x, y, z : \texttt{Nat}.\; x + (y + z) =_{\texttt{Nat}} (x + y) + z$
   b) $\forall x : \texttt{Nat}.\; x + \texttt{0} =_{\texttt{Nat}} x$
   c) $\forall x, y : \texttt{Nat}.\; x + \texttt{S}\,y =_{\texttt{Nat}} \texttt{S}\,(x + y)$
   d) $\forall x, y : \texttt{Nat}.\; x + y =_{\texttt{Nat}} y + x$

(3) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
   a) $\forall x : \texttt{Nat}.\; x \times \texttt{0} =_{\texttt{Nat}} \texttt{0}$
   b) $\forall x, y : \texttt{Nat}.\; x \times \texttt{S}\,y =_{\texttt{Nat}} (x \times y) + x$
   c) $\forall x, y : \texttt{Nat}.\; x \times y =_{\texttt{Nat}} y \times x$
   d) $\forall x, y, z : \texttt{Nat}.\; (x + y) \times z =_{\texttt{Nat}} x \times z + y \times z$
   e) $\forall x, y, z : \texttt{Nat}.\; x \times (y \times z) =_{\texttt{Nat}} (x \times y) \times z$

(4) Construir a partir del recursor `nat_rec` un término $\texttt{minus} : \texttt{Nat} \to \texttt{Nat} \to \texttt{Nat}$ que cumple las conversiones

$$\begin{array}{rcl} \texttt{minus}\,\texttt{0}\,n & \cong & \texttt{0} \\ \texttt{minus}\,(\texttt{S}\,n)\,\texttt{0} & \cong & \texttt{S}\,n \\ \texttt{minus}\,(\texttt{S}\,n)\,(\texttt{S}\,m) & \cong & \texttt{minus}\,n\,m \end{array}$$

En lo siguiente, se escribe $n \dot{-} m :\equiv \texttt{minus}\,n\,m$.

(5) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
   a) $\forall x : \texttt{Nat}.\; x \dot{-} 0 =_{\texttt{Nat}} x$
   b) $\forall x, y, z : \texttt{Nat}.\; (x + y) \dot{-} (x + z) =_{\texttt{Nat}} y - z$
   c) $\forall x, y : \texttt{Nat}.\; (x + y) \dot{-} y =_{\texttt{Nat}} x$

Ejercicio 2 (El orden y sus propiedades)**.** Se define el orden (usual) sobre los enteros naturales por $n \leq m :\equiv n \dot{-} m =_{\texttt{Nat}} \texttt{0}$ (usando la resta acotada definida en el ejercicio anterior).

(1) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
   a) $\forall x : \texttt{Nat}.\; x \leq x$
   b) $\forall x, y, z : \texttt{Nat}.\; x \leq y \to y \leq z \to x \leq z$
   c) $\forall x, y : \texttt{Nat}.\; x \leq y \to y \leq x \to x =_{\texttt{Nat}} y$

d) $\forall x, y : \mathtt{Nat}\,.\; x \leq y \vee y \leq x$
e) $\forall x, y : \mathtt{Nat}\,.\; x \leq y \to x \leq \mathtt{S}\, y$
f) $\forall x, y : \mathtt{Nat}\,.\; x \leq \mathtt{S}\, y \to x \leq y \vee x =_{\mathtt{Nat}} \mathtt{S}\, y$
g) $\forall x, y, z : \mathtt{Nat}\,.\; x \leq y \to z + x \leq z + y$

Se define el orden estricto sobre los enteros naturales por $n < m \;:\equiv\; \mathtt{S}\, n \leq m$.

(2) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
a) $\forall x, y : \mathtt{Nat}\,.\; x < y \to x \leq y$
b) $\forall x : \mathtt{Nat}\,.\; \neg(x < x)$
c) $\forall x : \mathtt{Nat}\,.\; \neg(x < 0)$
d) $\forall x, y, z : \mathtt{Nat}\,.\; x < y \to y \leq z \to x < z$
e) $\forall x, y, z : \mathtt{Nat}\,.\; x \leq y \to y < z \to x < z$
f) $\forall x, y, z : \mathtt{Nat}\,.\; x < y \to y < z \to x < z$
g) $\forall x, y : \mathtt{Nat}\,.\; x < y \to x < \mathtt{S}\, y$
h) $\forall x, y : \mathtt{Nat}\,.\; x < \mathtt{S}\, y \to x < y \vee x =_{\mathtt{Nat}} y$
i) $\forall x, y : \mathtt{Nat}\,.\; x < y \leftrightarrow x \leq y \wedge \neg(x =_{\mathtt{Nat}} y)$

(3) Demostrar (construyendo un término de prueba) el principio de inducción fuerte:

$$\forall P : \mathtt{Nat} \to \mathsf{Set}\,.\; (\forall x : \mathtt{Nat}\,.\; (\forall y : \mathtt{Nat}\,.\; y < x \to P\, y) \to P\, x) \;\to\; \forall x : \mathtt{Nat}\,.\; P\, x$$

Ejercicio 3 (Listas)**.** En la teoría de tipos de Martin-Löf, se define el tipo (polimórfico) de las listas mediante las constantes

$$\begin{array}{lcl}
\mathtt{List} & : & \mathsf{Set} \to \mathsf{Set} \\
\mathtt{nil} & : & \forall A : \mathsf{Set}\,.\; \mathtt{List}\, A \\
\mathtt{cons} & : & \forall A : \mathsf{Set}\,.\; A \to \mathtt{List}\, A \to \mathtt{List}\, A \\
\mathtt{list_rec} & : & \forall A : \mathsf{Set}\,.\; \forall X : \mathtt{List}\, A \to \mathsf{Set}\,. \\
 & & \quad X\,(\mathtt{nil}\, A) \to \\
 & & \quad (\forall x : A\,.\, \forall l : \mathtt{List}\, A\,.\; X\, l \to X\,(\mathtt{cons}\, A\, x\, l)) \to \\
 & & \quad \forall l : \mathtt{List}\, A\,.\, X\, l
\end{array}$$

con las δ-reglas: $\begin{array}{lcl} \mathtt{list\_rec}\, A\, X\, x\, f\,(\mathtt{nil}\, A) & \succ & x \\ \mathtt{list\_rec}\, A\, X\, x\, f\,(\mathtt{cons}\, A\, a\, l) & \succ & f\, a\, l\,(\mathtt{list\_rec}\, A\, X\, x\, f\, l) \end{array}$

(1) Construir a partir del recursor $\mathtt{list\_rec}$ términos $\mathtt{length} : \forall A : \mathsf{Set}\,.\, \mathtt{List}\, A \to \mathtt{Nat}$ y $\mathtt{concat} : \forall A : \mathsf{Set}\,.\, \mathtt{List}\, A \to \mathtt{List}\, A \to \mathtt{List}\, A$ que cumplen las conversiones

$$\begin{array}{rcl}
\mathtt{length}\, A\,(\mathtt{nil}\, A) & \cong & \mathtt{0} \\
\mathtt{length}\, A\,(\mathtt{cons}\, A\, a\, l) & \cong & \mathtt{S}\,(\mathtt{length}\, A\, l) \\
\\
\mathtt{concat}\, A\,(\mathtt{nil}\, A)\, l' & \cong & l' \\
\mathtt{concat}\, A\,(\mathtt{cons}\, A\, a\, l)\, l' & \cong & \mathtt{cons}\, A\, a\,(\mathtt{concat}\, A\, l\, l')
\end{array}$$

(2) Demostrar (construyendo términos de pruebas) las siguientes proposiciones:
a) $\forall A : \mathsf{Set}\,.\, \forall l, l' : \mathtt{List}\, A\,.\; \mathtt{length}\, A\,(\mathtt{concat}\, A\, l\, l') =_{\mathtt{Nat}} \mathtt{length}\, A\, l + \mathtt{length}\, A\, l'$
b) $\forall A : \mathsf{Set}\,.\, \forall l : \mathtt{List}\, A\,.\; \mathtt{concat}\, A\,(\mathtt{nil}\, A)\, l =_{\mathtt{List}\, A} l$
c) $\forall A : \mathsf{Set}\,.\, \forall l : \mathtt{List}\, A\,.\; \mathtt{concat}\, A\, l\,(\mathtt{nil}\, A) =_{\mathtt{List}\, A} l$
d) $\forall A : \mathsf{Set}\,.\, \forall l, l', l'' : \mathtt{List}\, A\,.\; \mathtt{concat}\, A\, l\,(\mathtt{concat}\, A\, l'\, l'') =_{\mathtt{List}\, A} \mathtt{concat}\, A\,(\mathtt{concat}\, A\, l\, l')\, l''$

Ejercicio 4 (Vectores)**.** En la teoría de tipos de Martin-Löf, se define el tipo (polimórfico) de los vectores (o listas dependientes) mediante las constantes

$$\begin{array}{lcl}
\mathtt{Vect} & : & \mathsf{Set} \to \mathtt{Nat} \to \mathsf{Set} \\
\mathtt{vnil} & : & \forall A : \mathsf{Set}\,.\ \mathtt{Vect}\,A\,\mathtt{0} \\
\mathtt{vcons} & : & \forall A : \mathsf{Set}\,.\,\forall n : \mathtt{Nat}\,.\ A \to \mathtt{Vect}\,A\,n \to \mathtt{Vect}\,A\,(\mathsf{S}\,n) \\
\mathtt{vect_rec} & : & \forall A : \mathsf{Set}\,.\ \forall X : (\forall n : \mathtt{Nat}\,.\,\mathtt{Vect}\,A\,n \to \mathsf{Set})\,. \\
 & & \quad X\,\mathtt{0}\,(\mathtt{vnil}\,A) \to \\
 & & \quad (\forall n : \mathtt{Nat}\,.\,\forall x : A\,.\,\forall v : \mathtt{Vect}\,A\,n\,.\ X\,n\,v \to X\,(\mathsf{S}\,n)\,(\mathtt{vcons}\,A\,n\,x\,v)) \to \\
 & & \quad \forall n : \mathtt{Nat}\,.\,\forall v : \mathtt{Vect}\,A\,n\,.\,X\,n\,v
\end{array}$$

con las δ-reglas:
$$\begin{array}{lcl}
\mathtt{vect_rec}\,A\,X\,x\,f\,\mathtt{0}\,(\mathtt{vnil}\,A) & \succ & x \\
\mathtt{vect_rec}\,A\,X\,x\,f\,(\mathsf{S}\,n)\,(\mathtt{vcons}\,A\,n\,a\,v) & \succ & f\,n\,a\,v\,(\mathtt{vect_rec}\,A\,X\,x\,f\,n\,v)
\end{array}$$

(1) Construir a partir del recursor `vect_rec` un término

$$\mathtt{vconcat} : \forall n : \mathtt{Nat}\,.\,\mathtt{Vect}\,A\,n \to \forall m : \mathtt{Nat}\,.\,\mathtt{Vect}\,A\,m \to \mathtt{Vect}\,A\,n\,(n+m)$$

que cumple las conversiones

$$\begin{array}{rcl}
\mathtt{vconcat}\,A\,\mathtt{0}\,(\mathtt{vnil}\,A)\,m\,w & \cong & w \\
\mathtt{vconcat}\,A\,(\mathsf{S}\,n)\,(\mathtt{vcons}\,A\,a\,v)\,m\,w & \cong & \mathtt{vcons}\,A\,a\,(m+n)\,(\mathtt{vconcat}\,A\,n\,v\,m\,w)
\end{array}$$

¿Cuáles conversiones tiene que cumplir la suma $n+m$ para que el término `vconcat` esté bien tipado? ¿La misma definición funcionaría con una suma $n+m$ definida por recursión sobre el segundo argumento?

(2) Explicar por qué no se puede construir un término de tipo

$$\begin{array}{l}
\forall n : \mathtt{Nat}\,.\,\forall u : \mathtt{Vect}\,A\,n\,.\,\forall m : \mathtt{Nat}\,.\,\forall v : \mathtt{Vect}\,A\,m\,.\,\forall p : \mathtt{Nat}\,.\,\forall w : \mathtt{Vect}\,A\,p\,. \\
\quad \mathtt{vconcat}\,A\,n\,u\,(m+p)\,(\mathtt{vconcat}\,A\,m\,v\,p\,w) = \mathtt{vconcat}\,A\,(n+m)\,(\mathtt{vconcat}\,A\,n\,u\,m\,v)\,p\,w
\end{array}$$

(3) Definir una función $\mathtt{list\_of\_vect} : \forall A : \mathsf{Set}\,.\,\forall n : \mathtt{Nat}\,.\ \mathtt{Vect}\,A\,n \to \mathtt{List}\,A$ que convierte cada vector en la correspondiente lista, y demostrar que

$$\forall A : \mathsf{Set}\,.\,\forall n : \mathtt{Nat}\,.\,\forall v : \mathtt{Vect}\,A\,n\,.\ \mathtt{length}\,A\,(\mathtt{list\_of\_vect}\,A\,n\,v) =_{\mathtt{Nat}} n$$

(4) Demostrar que

$$\begin{array}{l}
\forall A : \mathsf{Set}\,.\,\forall n : \mathtt{Nat}\,.\,\forall v : \mathtt{Vect}\,A\,n\,.\,\forall m : \mathtt{Nat}\,.\,\forall w : \mathtt{Vect}\,A\,m\,. \\
\quad \mathtt{list_of_vect}\,A\,(n+m)\,(\mathtt{vconcat}\,A\,n\,v\,m\,w) =_{\mathtt{List}\,A} \\
\quad \mathtt{concat}\,A\,(\mathtt{list_of_vect}\,A\,n\,v)\,(\mathtt{list_of_vect}\,A\,m\,w)
\end{array}$$

(5) Definir una función $\mathtt{vect\_of\_list} : \forall A : \mathsf{Set}\,.\,\forall l : \mathtt{List}\,A\,.\,\mathtt{Vect}\,A\,(\mathtt{length}\,A\,l)$ que convierte listas en vectores de misma longitud, y demostrar que

$$\forall A : \mathsf{Set}\,.\,\forall l : A\,.\ \mathtt{list\_of\_vect}\,A\,(\mathtt{length}\,A\,l)\,(\mathtt{vect\_of\_list}\,A\,l) =_{\mathtt{List}\,A} l$$

Ejercicio 5 (Axioma K de Streicher e igualdad de John Major)**.** En la teoría de tipos de Martin-Löf, la proposición

$$\forall A : \mathsf{Set}\,.\,\forall x, y : A\,.\,\forall e, e' : x =_A y\,.\ e =_{x=_A y} e'$$

(que expresa que, entre dos objetos $x, y : A$ cualesquiera, hay a lo sumo una prueba de igualdad) es indecidible, y los recientes trabajos sobre la teoría de tipos homotópica sugieren que es más interesante trabajar en teorías de tipos que refutan esta proposición. Sin embargo, en 1992 Streicher propuso extender la teoría de tipos de Martin-Löf con una nueva constante

$$\mathtt{K} \;:\; \forall A : \mathsf{Set} \,.\, \forall x : A \,.\, \forall P : (x =_A x \to \mathsf{Set}) \,.\, P\,(\mathtt{refl}\,A\,x) \to \forall e : x =_A x \,.\, P\,e$$

con la δ-regla $\quad \mathtt{K}\;A\;a\;P\;p\;(\mathtt{refl}\,A\,a) \;\succ\; p.$

(1) Demostrar que el axioma K de Streicher implica las siguientes fórmulas:
   a) $\forall A : \mathsf{Set} \,.\, \forall x : A \,.\, \forall e : x =_A x \,.\; e =_{x=_A x} \mathtt{refl}\,A\,x$
   b) $\forall A : \mathsf{Set} \,.\, \forall x : A \,.\, \forall e, e' : x =_A x \,.\; e =_{x=_A x} e'$
   c) $\forall A : \mathsf{Set} \,.\, \forall x, y : A \,.\, \forall e, e' : x =_A y \,.\; e =_{x=_A y} e'$

En 2002, McBride introdujo la igualdad heterogénea $x =_{A,B} y$ (con $x : A$ e $y : B$), llamada *igualdad de John Major*[1] y definida a partir de las constantes

$$\begin{aligned}
\mathtt{JMEq} \;&:\; \forall A : \mathsf{Set} \,.\, A \to \forall B : \mathsf{Set} \,.\, B \to \mathsf{Set} \qquad \text{(notación: } \mathtt{JMEq}\,A\,x\,B\,y \;\equiv\; x =_{A,B} y\text{)} \\
\mathtt{JMrefl} \;&:\; \forall A : \mathsf{Set} \,.\, \forall x : A \,.\; \mathtt{JMEq}\,A\,x\,A\,x \\
\mathtt{JMrec} \;&:\; \forall A : \mathsf{Set} \,.\, \forall x : A \,.\, \forall P : (\forall B : \mathsf{Set} \,.\, \forall y : B \,.\, \mathtt{JMEq}\,A\,x\,B\,y \to \mathsf{Set}) \,. \\
&\qquad P\,A\,x\,(\mathtt{JMrefl}\,A\,x) \to \forall B : \mathsf{Set} \,.\, \forall y : B \,.\, \forall e : \mathtt{JMEq}\,A\,x\,B\,y \,.\; P\,B\,y\,e
\end{aligned}$$

con la δ-regla: $\quad \mathtt{JMrec}\;A\;a\;P\;p\;A\;a\;(\mathtt{JMrefl}\,A\,a) \;\succ\; p.$

(2) Demostrar que la igualdad usual implica la igualdad de John Major:
$\forall A : \mathsf{Set} \,.\, \forall x, y : A \,.\; \mathtt{Eq}\,A\,x\,y \to \mathtt{JMEq}\,A\,x\,A\,y$

Se llama *axioma de John Major* a la proposición recíproca, que expresa que la igualdad de John Major implica la igualdad usual: $\quad \forall A : \mathsf{Set} \,.\, \forall x, y : A \,.\; \mathtt{JMEq}\,A\,x\,A\,y \to \mathtt{Eq}\,A\,x\,y$

(3) Demostrar que el axioma K de Streicher implica el axioma de John Major.

(4) Demostrar que el axioma de John Major implica el axioma K de Streicher.

[1] Basándose en la política de John Major (1943–), líder del Partido Conservador y primer ministro del Reino Unido de 1990 a 1997 (después de Margaret Thatcher y antes de Tony Blair).