Programación Funcional

Trabajo Práctico Nro. 12

Temas: λ-cálculo: programación con λ-cálculo. sustitución, α-equivalencia, β-reducción,

1. Para la representación de números naturales como numerales de Church:

   a) Dar λ-expresiones que representen a *mult* y *potencia*.

   b) Dar una λ-expresión que represente a *isNotZero*.

   c) ⍟ Dar una λ-expresión que represente a *pred*.
   Sugerencia: Dar antes una λ-expresión que represente a *prefn*, definida por:

   $$\begin{array}{l} \mathit{prefn}\ f\ (\underline{\mathit{true}}, x) = (\underline{\mathit{false}}, x) \\ \mathit{prefn}\ f\ (\underline{\mathit{false}}, x) = (\underline{\mathit{false}}, f x) \end{array}$$

   d) Dar una λ-expresión que represente a *resta*, definida por:

   $$\begin{array}{rll} \mathit{resta}\ \ \underline{m}\ \ \underline{n} & = \underline{m-n} & \text{si } m \geq n \\ & = \underline{0} & \text{si } m < n \end{array}$$

   Ayuda: Asumir definido *pred*.

2. Dar λ-expresiones que representen las funciones *isempty*, *head* y *tail*, para las definiciones de listas vistas en teoría.

3. *a*) Definir con λ-expresiones una representación apropiada para árboles `TipTree`:

   1) Representando los árboles explícitamente.
   2) ⍟ Representando los árboles mediante su patrón de recursión.

   b) Para ambas representaciones, definir en λ-cálculo las siguientes operaciones:

   - *leaf*: construye una hoja del árbol.
   - *tree*: construye un nodo del árbol.
   - *isNotLeaf*: devuelve True sii el árbol fue construido con la operación *tree*.
   - *left*: devuelve el subárbol izquierdo de un árbol no vacío.
   - *right*: devuelve el subárbol derecho de un árbol no vacío.

4. *a*) Indicar, para cada variable, cuáles de sus ocurrencias son libres y cuáles ligadas, en las siguientes expresiones. En caso de ser una ocurrencia ligada, indicar a qué binder lo está.

1) (λy.x(λx.x) z)
2) (λy.y(λx.y) yx)
3) (λxx.xy(λyy.xy)) x
4) (λxyz.(λy.yz) w)(λx.xy) wz

5. *a*) Indicar cuáles de los siguientes pares de λ-expresiones son α-equivalentes y cuáles no lo son. Justificar la respuesta.

    1) (λxyz.x(λy.yz) w)   (λtuv.t(λz.zv)) w
    2) (λxyz.x(λy.yz) w)   (λxyw.x(λy.yw) z)
    3) (λxyz.x(λy.yz) w)   (λvut.v(λv.vt) x)
    4) (λxyz.x(λy.yz) w)   (λxtz.x(λu.tz)) w

   b) Escriba en Haskell una función que dados dos términos lambda decida si son α-equivalentes o no.

   c) Utilice la función del inciso anterior para verificar las respuestas del primero.

Ejercicios complementarios

6. Dadas las siguientes representaciones **alternativas** de números naturales en λ-cálculo, definir las funciones *succ* y *pred*, y los predicados *iszero* e *isnotzero*.

   a) $\underline{0} = \lambda x.\underline{\mathit{false}}$
   $\underline{n+1} = \underline{\mathit{pair}}\ \ \underline{\mathit{true}}\ \ \underline{n}$

   b) $\underline{0} = \lambda x.\underline{\mathit{true}}$
   $\underline{n+1} = \underline{\mathit{pair}}\ \ \underline{n}\ \ \underline{\mathit{false}}$

7. Demostrar que para toda variable x y λ-términos M y N, si x no ocurre libre en M entonces $M\{x \leftarrow N\} =_\alpha M$.

8. El grafo de reducción de un término M consiste en el conjunto de nodos $\{N|M \to^* N\}$ y en las aristas $\{(M,N)|M \to^* N\}$.

   a) construya el grafo de reducción de $(\lambda x.\lambda y.x)z\Omega$, donde $\Omega =_{\mathrm{def}} (\lambda x.xx)(\lambda x.xx)$

   b) construya el grafo de reducción de Ω

   c) exhiba un término cuyo grafo de reducción (donde los nodos se representan con "$*$") sea $* \to * \to * \to \dots$