Lenguajes de programación 2020-1
Boletín de ejercicios 2

Favio E. Miranda Perea   Javier Enríquez Mendoza

20 de septiembre de 2019

Semántica

1. Se requiere extender al lenguaje EAB con árboles binarios con etiquetas en un tipo dado S como sigue:

   - Tipo de árboles binarios: $\mathsf{Tree\ S}$
   - Expresiones en sintaxis concreta :
     - void para el árbol vacío
     - $\mathsf{tree(a,l,r)}$ para construir el árbol con raiz a y subárboles izquierdo l y derecho r.
     - $\mathsf{tcase\ t\ of}\ \{\mathsf{void} \Rightarrow \mathsf{e_1} \mid \mathsf{tree(x,y,z)} \Rightarrow \mathsf{e_2}\}$, que realiza un análisis de casos de acuerdo a si t es el árbol vacío o no. Obsérvese que en tal expresión x, y, z son variables ligadas en e_2.

   Realizar lo siguiente:

   a) Definir la sintaxis abstracta para esta extensión

   b) Definir la semántica operacional para esta extensión

   c) Definir la semántica estática para esta extensión. La estrategia es la usual (llamada por valor) por lo que se debe definir claramente qué expresiones son valores.

   d) Definir dentro de la extensión las funciones $\mathsf{root,lst,rst}$ que devuelven la raiz, subárbol izquierdo y subárbol derecho respectivamente.

Cálculo Lambda

2. Analiza cada una de las siguientes expresiones lambda. Si la expresión es una función, identifica la variable ligada, el cuerpo de la función y analízalo. Si la expresión es una aplicación, identifica la función a aplicar y el argumento de la función y analízalos de la misma manera. En todos los casos eliminar los paréntesis innecesarios.

   a) $\lambda a.(a\ \lambda b.(b\ a))$

   b) $\lambda x.\lambda y.\lambda z.((z\ x)\ (z\ y))$

c) $(\lambda f.\lambda g.(\lambda h.(g\ h)\ f)\ \lambda p.\lambda q.p)$

d) $\lambda x.\lambda y.\lambda z.\lambda w.(w\ (z\ (y\ x)))$

e) $(\lambda p.(\lambda q.p\ \lambda x.(x\ p))\ \lambda i.\lambda j.(j\ i))$

3. Evalúa las siguientes expresiones lambda. En todos los casos eliminar los paréntesis innecesarios

   a) $((\lambda x.\lambda y.(y\ x))\ \lambda p.\lambda q.p)\ \lambda i.i)$

   b) $(((\lambda x.\lambda y.\lambda z.((x\ y)\ z)\ \lambda f.\lambda a.(f\ a))\ \lambda i.i)\ \lambda j.j)$

   c) $(\lambda h.((\lambda a.\lambda f.(f\ a)\ h)\ h)\ \lambda f.(f\ f))$

   d) $((\lambda p.\lambda q.(p\ q)\ (\lambda x.x\ \lambda a.\lambda b.a))\ \lambda k.k)$

   e) $(((\lambda f.\lambda g.\lambda x.(f\ (g\ x))\ \lambda s.(s\ s))\ \lambda a.\lambda b.b)\ \lambda x.\lambda y.x)$

4. Identifica las variables libres y ligadas de las siguientes expresiones:

   a) $\lambda x.\lambda y.(\lambda x.y\ \lambda y.x)$

   b) $\lambda x.(x\ (\lambda y.(\lambda x.x\ y)\ x))$

   c) $\lambda a.(\lambda b.a\ \lambda b.(\lambda a.a\ b))$

   d) $(\lambda f.b\ \lambda b.(\lambda f.f\ b))$

   e) $\lambda p.\lambda q.(\lambda r.(p\ (\lambda q.(\lambda p.(r\ p))))\ (q\ p))$

5. En cada uno de los incisos del ejercicio anterior aplicar α-equivalencias para obtener términos donde todos los ligados sean a variables distintas.

6. Considérese el siguiente término M:

$$M =_{def} \lambda x.(\lambda z.zx)((\lambda v \lambda w.wv)yf)$$

   a) Identifique los dos redex que hay en M, digamos R_1 y R_2.

   b) Calcule los reductos N_1 de R_1 y N_2 de R_2.

   c) Halle la forma normal V de M.

7. Calcular las formas normales de los siguientes terminos:

   a) $(\lambda xy.yx)\, true\, (\lambda x.x)$

   b) $(\lambda xy.y(yx))\, false\, (\lambda x.x)$

   c) $(\lambda xy.y(yx))\, ((\lambda z.true) false)\, (\lambda x.x)$

   d) $(\lambda xyz.xz(yz))\, (\lambda s.\lambda t.s)\, (\lambda u.u)w$

8. Reducir las siguientes expresiones a su forma normal o mostrar que dicha forma no existe.

   a) PI donde $P :=_{def} (\lambda x.xxy)I$, con $I :=_{def} \lambda x.x$

   b) $Y :=_{def} \lambda f.QQ$, donde $Q :=_{def} \lambda x.f(xx)$

c) $L :=_{def} RR$ donde $R :=_{def} (\lambda x.xxy)$

d) $(\lambda x.xL)M$ donde $M :=_{def} \lambda x.y$

e) $\big((\lambda u \lambda v.Tv)y\big)W$ donde $T :=_{def} \lambda w.w(\lambda x.xu)$, $W :=_{def} \lambda z.\lambda y.zy$

9. Reducir las siguientes expresiones a su forma normal o en su caso mostrar que dicha forma no existe, los términos I, K y S son como en el siguiente ejercicio.

   - $\lambda x.xK\Omega \quad$ donde $\Omega = (\lambda x.xx)(\lambda x.xx)$
   - $(\lambda x.x(II))z$
   - $\lambda x.\lambda y.y(yx)\,\mathsf{false}(\lambda x.x)$
   - $(\lambda u.\lambda v.(\lambda w.w(\lambda x.xu))v)y(\lambda z.\lambda y.zy)$
   - $S(KI)(KI)$

10. Con respecto a las funciones sucesor y predecesor en numerales de Church realice lo siguiente:

    a) Evalúe $\mathsf{suc}(\mathsf{pred}\,\bar{0})$

    b) Evalúe $\mathsf{suc}(\mathsf{pred}\,\bar{1})$

    c) ¿ A qué se evalúa $\mathsf{suc}(\mathsf{pred}\,\bar{n})$ para un numeral arbitrario $\bar{n}$?

    d) ¿ Qué se puede decir de la evaluación de $\mathsf{suc}(\mathsf{pred}\,r)$ para un término arbitrario r ?

    e) Conteste todos los incisos anteriores para $\mathsf{pred}(\mathsf{suc}\,x)$

11. Implemente las siguientes funciones para numerales de Church:

    a) Relación de orden `lq`

    b) Igualdad de numerales `eq`

    c) Diferencia positiva `dp` (si $n < m$, `dp` n m debe devolver 0)

    d) Cociente y residuo entre dos numerales.

12. Determinar a que operación corresponde la siguiente expresión M al ser aplicada a un numeral de Church $\bar{n}$, donde las operaciones $\mathsf{pair}, \mathsf{iszero}, \mathsf{fst}, \mathsf{snd}$ son las definidas en la nota de clase 6.

$$M =_{def} \lambda n \lambda f \lambda x.\,\mathsf{snd}(nH(\mathsf{pair}1\ x))$$

donde

$$H =_{def} \lambda p.(\mathsf{if}\ (\mathsf{iszero}(\mathsf{fst}\,p))\ \mathsf{then}\ \mathsf{pair}\ 1\ (f(\mathsf{snd}\,p))\ \mathsf{else}\ \mathsf{pair}\ 0\ (\mathsf{snd}\,p))$$

13. Una manera de implementar listas en el cálculo lambda es mediante pares, la lista $x : xs$ se implementa mediante el par $\langle x, xs\rangle$. De esta manera se tiene que

$$[x_1, \ldots, x_k] =_{def} \langle x_1, \langle x_2, \ldots, \langle x_n, Nil\rangle \ldots\rangle\rangle$$

Con esta idea en mente implemente las siguientes funciones:

a) Lista vacía: `nil`

b) Constructor de listas `cons`

c) Cabeza: `head`

d) Cola: `tail`

e) Test de lista vacía: `isnil`

14. Otra manera de implementar listas en el cálculo lambda es mediante la función de iteración fold, la idea es representar una lista $[x_1, \ldots, x_n]$ como:

$$[a_1, \ldots, a_n] =_{def} \lambda f.\lambda x.fa_1(fa_2(\ldots fa_{n-1}(fa_n x)\ldots))$$

a) Explique la idea de esta implementación en analogía a los numerales de Church.

b) Implemente las listas $[1], [2, 3], [4, 5, 6]$

c) Implemente las siguientes funciones:

1) Lista vacía: `nil`
2) Constructor de listas `cons`
3) Cabeza: `head`
4) Cola: `tail` (esta corresponde al predecesor por lo que resulta más complicada que las demás)
5) Test de lista vacía: `isnil`

15. Definir en el cálculo lambda puro la función `iseven` que toma un natural y decide si es par. Verifique su definición evaluando las expresiones `iseven` $\bar{1}$ e `iseven` $\bar{2}$ (puede omitir algunos pasos usando $\rightarrow^{\star}$). Explique la idea utilizada en su definición

16. Muestre que los siguientes términos son combinadores de punto fijo

a) El operador V de Turing (ver nota 6)

b) El operador K de Klop (ver nota 6)

c) El operador $Z = QQ$ donde $Q = \lambda x \lambda y.y(\lambda z.xxyz)$

17. Defina mediante el uso del combinador de punto fijolas siguientes funciones recursivas:

- Exponenciación de naturales.
- Predecesor de naturales.
- Concatenación y reversa de listas.

18. Defina mediante el uso del combinador de punto fijo las siguientes funciones recursivas:

```
even x = if iszero x then true else
                  if iszero (pred x) then false else
                       even (pred (pred x))

mq 0 (suc x) = true
mq (suc x) (suc y) = mq x y
```

Tipos producto y suma

19. Defina las siguientes funciones que involucran tipos producto:

    a) `revpair` tal que `revpair` $\langle e_1, e_2\rangle \to^{\star} \langle e_2, e_1\rangle$
    b) `convtriple` tal que `convtriple` $\langle e_1, \langle e_2, e_3\rangle\rangle \to^{\star} \langle\langle e1, e_2\rangle, e_3\rangle$

    Muestre que sus definiciones son correctas mediante el tipado y la evaluación de algunos casos sencillos.

20. En teoría de conjuntos se cumple la siguiente ley distributiva:

    $$A \times (B \cup C) = (A \times B) \cup (A \times C)$$

    Muestre que ésta también es válida para tipos suma y producto definiendo dos funciones:

    - `pts`: $\mathsf{S} \times (\mathsf{T} + \mathsf{R}) \to (\mathsf{S} \times \mathsf{T}) + (\mathsf{S} \times \mathsf{R})$
    - `stp`: $(\mathsf{S} \times \mathsf{T}) + (\mathsf{S} \times \mathsf{R}) \to \mathsf{S} \times (\mathsf{T} + \mathsf{R})$

    ¿A qué debe corresponder la composición de funciones `pts` $\circ$ `stp` y `stp` $\circ$ `pts` ?

21. Chon Hacker afirma que los tipos suma son innecesarios puesto que se pueden implementar con productos. Su idea es que las etiquetas $\mathsf{inl}, \mathsf{inr}$ que sirven para distinguir el origen de las expresiones de tipo suma son esencialmente valores booleanos por lo que expresiones como $\mathsf{inl}\,5$ o inr "*sapo*" pueden representarse mediante los pares $\langle \mathsf{false}, 5\rangle$ y $\langle \mathsf{true}, \text{"}\textit{sapo}\text{"}\rangle$.

    - Defina al tipo suma $\mathsf{S} + \mathsf{T}$ y sus operaciones siguiendo la idea anterior.
    - Discuta las ventajas y desventajas de esta definición.

Preguntas de exámenes anteriores

22. Yo Lambda desea extender el lenguaje EAB con un nuevo operador match para definir expresiones mediante un análisis de casos sobre números (cero y sucesor).

    - Sintaxis: $\mathsf{match}\ e_1\ \mathsf{with}\ 0 \Rightarrow\ e_2\ \mid suc\ x\ \Rightarrow e_3\ \mathsf{end}$
    - Semántica: Se evalúa e_1 a un valor v, si $v = 0$ se devuelve e_2 y si $v = suc\ x$ se devuelve e_3.

    Realiza lo siguiente:

    a) Define la sintaxis abstracta para match. Nota que hay un ligado explícito en e_3
    b) Define la semántica operacional para esta extensión.
    c) Define la semántica estática para esta extensión.
    d) Sea $e =_{def} \mathsf{match}\ 1\ \mathsf{with}\ 0 \Rightarrow\ true\ \mid suc\ x\ \Rightarrow \mathsf{iszero}\ x\ \mathsf{end}$.
    Muestra que:

i) $\vdash e : \mathsf{Bool}$

ii) $e \to^\star bool[true]$

e) ¿Cómo definirias el operador `match` mediante azúcar sintáctica?

23. Sea $C =_{def} \lambda x.\lambda y.\lambda z.x(yz)$.

    a) Muestre que $C\bar{1}\bar{0} \to^\star \bar{0}$ donde $\bar{1}, \bar{0}$ son los numerales de Church para 0 y 1.

    b) Hallar la forma normal para $C\bar{1}\bar{2}$

    c) ¿Qué hace C en general con cualesquiera dos numerales?

24. Considere la siguiente función misteriosa:

```
mist x y = if iszero y then x else mist (suc x) (pred y)
```

    a) Defina a `mist` con el combinador de punto fijo Y.

    b) Evalúe `mist 2 3` usando la definición del inciso anterior. Puede omitir varios pasos pero no los que involucren al combinador Y.

    c) Proponga un nombre más adecuado para `mist`

25. Considere las siguientes definiciones

$$O =_{def} \lambda f.\lambda z.z \quad S =_{def} \lambda x.\lambda f.\lambda z.fx$$

    El numeral de Scott $\tilde{n}$ se define a partir de S y O de la manera usual, es decir, $\tilde{n} =_{def} S(S \ldots (S\ O) \ldots)$ con n presencias de S.

    a) Hallar la forma normal de $\tilde{2}$ y $\tilde{3}$.

    b) Sea $P =_{def} \lambda x.x\, I\, O$, donde I es la función identidad. Muestre que P implementa a la función predecesor para numerales de Scott. *Sugerencia*: observe que $\widetilde{n+1} = S\,\tilde{n}$

26. (2 pts) Definir en el cálculo lambda puro la función `iseven` que toma un natural y decide si es par. Verifique su definición evaluando las expresiones `iseven` $\bar{1}$ e `iseven` $\bar{2}$ (puede omitir algunos pasos usando $\to^\star$). Explique la idea utilizada en su definición
    Sugerencia: Iterar una funcion adecuada, la cual puede suponer definida.

¡Suerte!