Lenguajes y Compiladores. Práctico 4 del 01/04/2020

Objetivos: Comprender las ecuaciones semánticas para el lenguaje imperativo simple. Utilizar semántica denotacional para comparar si dos programas son equivalentes o no. Computar las cadenas de sucesivas ejecuciones de un ciclo, identificar el caso general de la cadena y probar por inducción que es correcto. Definir nuevas construcciones para el lenguaje imperativo, dando sus ecuaciones semánticas. Relacionar valores del dominio semántico $\Sigma \to \Sigma_\perp$ con la denotación de programas. Conocer los teoremas de sustitución, renombre y coincidencia y utilizarlos para concluir si dos programas son equivalentes o no.
Recordar que utilizamos tipografía sans-serif (como en x) para variables concretas y tipografía serif (como en e) para meta-variables.

Repaso. Decida si los siguiente argumentos son correctos:

(1) Sea D un dominio y $f\colon D \to D$. Si f es continua, entonces f tiene un menor punto fijo.
(2) Sea D un dominio y $f\colon D \to D$. Si f es continua, entonces $\perp$ es el menor punto fijo de f.
(3) Sea D un dominio y sea $f_1 \sqsubseteq f_2 \sqsubseteq f_3 \sqsubseteq \ldots$ una cadena en $D \to D$. Si el supremo de la cadena pertenece a la cadena, entonces existe un índice k tal que $f_k = f_{k+1} = f_{k+2}$.

Ejercicios.

(1) Utilizar la semántica denotacional para demostrar o refutar las siguientes equivalencias:
   (a) $c; \mathbf{skip} \equiv c$
   (b) $c_1; (c_2; c_3) \equiv (c_1; c_2); c_3$
   (c) $(\mathbf{if}\ b\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1); c_2 \equiv \mathbf{if}\ b\ \mathbf{then}\ c_0; c_2\ \mathbf{else}\ c_1; c_2$
   (d) $c_2; (\mathbf{if}\ b\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1) \equiv \mathbf{if}\ b\ \mathbf{then}\ c_2; c_0\ \mathbf{else}\ c_2; c_1$
   (e) $x := y\ ;\ z := w \equiv z := w\ ;\ x := y$

(2) Utilizar la semántica denotacional para demostrar o refutar las siguientes equivalencias:
   (a) $\mathbf{newvar}\ \mathsf{x} := e\ \mathbf{in}\ \mathbf{skip} \equiv \mathbf{skip}$
   (b) $\mathbf{newvar}\ \mathsf{x} := e\ \mathbf{in}\ \mathsf{y} := \mathsf{x} \equiv \mathsf{y} := e$
   (c) $\mathbf{newvar}\ \mathsf{x} := e_1\ \mathbf{in}\ (\mathbf{newvar}\ \mathsf{y} := e_2\ \mathbf{in}\ c) \equiv \mathbf{newvar}\ \mathsf{y} := e_2\ \mathbf{in}\ (\mathbf{newvar}\ \mathsf{x} := e_1\ \mathbf{in}\ c)$

(3) Teniendo en cuenta los ejercicios anteriores, discuta en grupo las siguientes afirmaciones:
   (a) El parser puede eliminar toda ocurrencia de **skip**.
   (b) El parser puede elegir inclinar las secuencias de más de dos comandos hacia la derecha o hacia la izquierda.

(4) Considere el comando $\mathbf{while\ true\ do}\ \ x := x - 1$
   (a) Dar la función F que define su semántica. Calcular la expresión más sencilla que pueda para F.
   (b) Existe algún n tal que $F^n \perp_{\Sigma \to \Sigma_\perp}$ no sea idénticamente $\perp$?
   (c) Considere la cadena en $\Sigma \to \Sigma_\perp$ dada por

$$\omega_i\ \sigma = \begin{cases} \sigma & \text{si } 0 \leq \sigma x \leq i \\ \perp & \text{caso contrario} \end{cases}$$

   Es sabido que la continuidad de F garantiza la igualdad $F\,(\bigsqcup \omega_i) = \bigsqcup F\omega_i$. Compruebe la misma calculando cada miembro de la igualdad para el caso de la cadena dada.

(5) Calcule la semántica denotacional de los siguientes comandos:
   (a) **while** $\mathsf{x} < 2$ **do if** $\mathsf{x} < 0$ **then** $\mathsf{x} := 0$ **else** $\mathsf{x} := \mathsf{x} + 1$
   (b) **while** $\mathsf{x} < 2$ **do if** $\mathsf{y} = 0$ **then** $\mathsf{x} := \mathsf{x} + 1$ **else skip**

(6) Suponga que $[\![\textbf{while } b \textbf{ do } c]\!]\sigma \neq \bot$.
   (a) Demuestre que existe $n \geq 0$ tal que $F^n \perp \sigma \neq \bot$.
   (b) Demuestre que si $\sigma' = [\![\textbf{while } b \textbf{ do } c]\!]\sigma$ entonces $\neg[\![b]\!]\sigma'$

(7) Demostrar o refutar las siguientes equivalencias usando semántica denotacional:
   (a) **while false do** $c \equiv$ **skip**
   (b) **while** b **do** $c \equiv$ **while** b **do** $(c; c)$
   (c) (**while** b **do** c) ; **if** b **then** c_0 **else** $c_1 \equiv$ (**while** b **do** c) ; c_1

(8) Considerar las siguientes definiciones como syntactic sugar del comando **for** $v := e_0$ **to** e_1 **do** c:
   (a) $v := e_0$; **while** $v \leq e_1$ **do** $c; v := v + 1$.
   (b) **newvar** $v := e_0$ **in while** $v \leq e_1$ **do** c; $v := v + 1$.
   (c) **newvar** $w := e_1$ **in newvar** $v := e_0$ **in while** $v \leq w$ **do** $c; v := v + 1$

   ¿Hay alguna que pueda considerarse satisfactoria? Justificar.

(9) Enunciar de manera completa el teorema de coincidencia y demostrar el caso **while**.

(10) Usando el Teorema de coincidencia para comandos probar que para todo par de comandos c_0, c_1, si
$$FV\ c_0 \cap FA\ c_1 = FV\ c_1 \cap FA\ c_0 = \emptyset,$$
entonces $[\![c_0; c_1]\!] = [\![c_1; c_0]\!]$.

(11) Considere los siguientes comandos:
$$\begin{aligned} c_0 &\doteq \textbf{newvar } \mathsf{x} := \mathsf{x} + \mathsf{y} \textbf{ in} \\ &\qquad c; \textbf{while } \mathsf{x} > 0 \textbf{ do } \mathsf{y} := \mathsf{y} + 1;\ \mathsf{x} := \mathsf{x} - 1 \\ c_1 &\doteq \textbf{newvar } \mathsf{y} := \mathsf{x} + \mathsf{z} \textbf{ in} \\ &\qquad c; \textbf{while } \mathsf{y} > 0 \textbf{ do } \mathsf{z} := \mathsf{z} + 1;\ \mathsf{y} := \mathsf{y} - 1 \end{aligned}$$
Asuma que c es un comando que satisface $(FV\ c) \cap \{\mathsf{x}, \mathsf{y}, \mathsf{z}\} = \emptyset$. Formule la relación que existe entre estos comandos (vistos como funciones que transforman estados), y pruebe tal relación sin calcular semántica.

(12) Considere la siguiente cadena en $\Sigma \to \Sigma_\perp$. Decida si existe un programa cuya semántica sea el supremo de la cadena. $f_i\,\sigma = \begin{cases} \sigma & \text{si } \sigma\,\mathsf{x} \leqslant \sigma\,\mathsf{y} \\ \bot & \text{en caso contrario} \end{cases}$