Lenguajes y Compiladores. Guía 5 del 12 de abril de 2016

Objetivos: Comprender las modificaciones necesarias que se deben realizar al dominio semántico para poder extender el lenguaje con un mecanismo de fallas y manejo de fallas. Poder construir trazas (i.e. derivaciones) de cómo se ejecuta un programa utilizando la semántica operacional. Poder relacionar la semántica denotacional con la operacional probando su equivalencia.

Recordar que los símbolos en cursiva representan metavariables. Por ejemplo, si en un ejercicio involucra los símbolos v y w, debe considerarse el caso en que ambas metavariables denoten la misma variable concreta.

Repaso.

(1) Escriba un teorema de coincidencia para la semántica operacional.
(2) Después de terminar con los ejercicios, pruebe el teorema de coincidencia (sin hacer ninguna inducción).

Ejercicios.

(1) Dado el programa $P \equiv$

$$\textbf{newvar } \mathsf{x} := \mathsf{y} + \mathsf{x} \textbf{ in}$$
$$\quad \textbf{while } \mathsf{x} > 0 \textbf{ do } !\mathsf{x};\ \textbf{if } \mathsf{x} > 0 \textbf{ then skip else fail}$$

Caracterice (sin calcular la semántica) los estados σ en los cuales P se comporta como **skip**.

(2) Demostrar o refutar las siguientes equivalencias usando semántica denotacional:
   (a) $c; \textbf{while true do skip} \ \equiv\ \textbf{while true do skip}$.
   (b) $c; \textbf{fail} \equiv \textbf{fail}$.
   (c) $\textbf{newvar } v := e \textbf{ in } v := v + 1; \textbf{fail} \equiv$
   $$\textbf{newvar } w := e \textbf{ in } w := w + 1; \textbf{fail}.$$
   (d) $\textbf{while } b \textbf{ do fail} \equiv \textbf{if } b \textbf{ then fail else skip}$.
   (e) $\mathsf{x} := 0; \textbf{catch } \mathsf{x} := 1 \textbf{ in while } \mathsf{x} < 1 \textbf{ do fail} \equiv$
   $$\mathsf{x} := 0; \textbf{while } \mathsf{x} < 1 \textbf{ do catch } \mathsf{x} := 1 \textbf{ in fail}$$

Considere el punto (a) para el lenguaje sin fallas: ¿la respuesta es la misma que para el lenguaje con fallas?

(3) Dados los programas
   (1) $\mathsf{y} := \mathsf{x} + \mathsf{y};\ \textbf{if } \mathsf{y} > 0 \textbf{ then } \mathsf{x} := \mathsf{x} - 1 \textbf{ else skip}$
   (2) $\textbf{while } \mathsf{x} > 0 \textbf{ do}$
   $$\quad \mathsf{y} := \mathsf{x} + \mathsf{y};\ \textbf{if } \mathsf{y} > 0 \textbf{ then } \mathsf{x} := \mathsf{x} - 1 \textbf{ else skip}$$

Computar la semántica operacional en estados σ tales que $\sigma\mathsf{x} = 2, \sigma\mathsf{y} = 1$.

(4) Computar la semántica operacional del siguiente programa en un estado arbitrario

newvar $\mathsf{x} := 2$ **in**
  while $\mathsf{x} > 0$ **do**
    $\mathsf{y} := \mathsf{x} + \mathsf{y}$; **if** $\mathsf{y} > 0$ **then** $\mathsf{x} := \mathsf{x} - 1$ **else skip**

(5) Pruebe:
  (a) Si $\langle c_0, \sigma\rangle \to^* \sigma'$, entonces $\langle c_0; c_1, \sigma\rangle \to^* \langle c_1, \sigma'\rangle$
  (b) Si $\langle c, [\sigma|\mathsf{x} : [\![e]\!]\sigma]\rangle \to^* \sigma'$, entonces

$$\langle \textbf{newvar } \mathsf{x} := e \textbf{ in } c, \sigma\rangle \to^* [\sigma'|\mathsf{x} : \sigma\mathsf{x}].$$

  (c) Si $\langle c, [\sigma|\mathsf{x} : [\![e]\!]\sigma]\rangle \to^* \langle c', \sigma'\rangle$, entonces

$$\langle \textbf{newvar } \mathsf{x} := e \textbf{ in } c, \sigma\rangle \to^* \langle \textbf{newvar } \mathsf{x} := \sigma' x \textbf{ in } c', [\sigma'|\mathsf{x} : \sigma x]\rangle$$

(6) Demostrar la corrección de la semántica operacional one-step (incluyendo **fail**) respecto de la semántica denotacional, demostrando simultáneamente
  (a) $\langle c, \sigma\rangle \to \sigma' \Longrightarrow [\![c]\!]\sigma = \sigma'$.
  (b) $\langle c, \sigma\rangle \to \langle \textbf{abort}, \sigma'\rangle \Longrightarrow [\![c]\!]\sigma = \langle \textbf{abort}, \sigma'\rangle$.
  (c) $\langle c, \sigma\rangle \to \langle c', \sigma'\rangle \implies [\![c]\!]\sigma = [\![c']\!]\sigma'$.

  Use inducción en la derivación.

(7) Demostrar simultáneamente:
  (a) $\langle c, \sigma\rangle \to^* \sigma' \Longleftarrow [\![c]\!]\sigma = \sigma'$.
  (b) $\langle c, \sigma\rangle \to^* \langle \textbf{abort}, \sigma'\rangle \Longleftarrow [\![c]\!]\sigma = \langle \textbf{abort}, \sigma'\rangle$.

  Use inducción en la estructura de c. Tendrá que usar los resultados probados en el ejercicio 5. El caso **while** requiere a su vez una inducción en el mínimo n tal que $F^n(\bot)\sigma \neq \bot$.

(8) Demostrar la equivalencia entre la semántica denotacional y la operacional del lenguaje imperativo simple incluyendo **fail**. Debe salir inmediatamente de los dos ejercicios anteriores.

(9) (opcional) Proponga una semántica operacional big-step $\Rightarrow \subseteq \Gamma_n \times \Gamma_t$ para el lenguaje LIS, tal que

$$\langle c, \sigma\rangle \Rightarrow \sigma' \quad \text{sii} \quad \langle c, \sigma\rangle \to^* \sigma' \ .$$

  Discutan cómo probarían la equivalencia; prueben la equivalencia en grupo (quienes hacen una dirección, no hacen la otra; y viceversa)