

## Lenguajes y Compiladores. Guía 5 del 18 de abril de 2017

**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  $P \equiv$

```
newvar  $x := y + x$  in  
  while  $x > 0$  do ! $x$ ; if  $x > 0$  then skip else fail
```

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

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

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

- (3) Computar la semántica operacional en estados  $\sigma$  tales que  $\sigma x = 2, \sigma y = 1$ . de los siguientes programas
  - (1)  $y := x + y$ ; **if**  $y > 0$  **then**  $x := x - 1$  **else** **skip**
  - (2) **while**  $x > 0$  **do**  
 $y := x + y$ ; **if**  $y > 0$  **then**  $x := x - 1$  **else** **skip**

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

```

newvar x := 2 in
  while x > 0 do
    y := x + y; if y > 0 then x := x - 1 else skip

```

- (5) Pruebe:

(a) Si  $\langle c_0, \sigma \rangle \rightarrow^* \sigma'$ , entonces  $\langle c_0; c_1, \sigma \rangle \rightarrow^* \langle c_1, \sigma' \rangle$

(b) Si  $\langle c, [\sigma|x : \llbracket e \rrbracket \sigma] \rangle \rightarrow^* \sigma'$ , entonces

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

(c) Si  $\langle c, [\sigma|x : \llbracket e \rrbracket \sigma] \rangle \rightarrow^* \langle c', \sigma' \rangle$ , entonces

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

- (6) Demostrar la corrección de la semántica operacional one-step (incluyendo **fail**) respecto de la semántica denotacional, demostrando simultáneamente

(a)  $\langle c, \sigma \rangle \rightarrow \sigma' \implies \llbracket c \rrbracket \sigma = \sigma'$ .

(b)  $\langle c, \sigma \rangle \rightarrow \langle \mathbf{abort}, \sigma' \rangle \implies \llbracket c \rrbracket \sigma = \langle \mathbf{abort}, \sigma' \rangle$ .

(c)  $\langle c, \sigma \rangle \rightarrow \langle c', \sigma' \rangle \implies \llbracket c \rrbracket \sigma = \llbracket c' \rrbracket \sigma'$ .

Use inducción en la derivación.

- (7) Demostrar simultáneamente:

(a)  $\langle c, \sigma \rangle \rightarrow^* \sigma' \iff \llbracket c \rrbracket \sigma = \sigma'$ .

(b)  $\langle c, \sigma \rangle \rightarrow^* \langle \mathbf{abort}, \sigma' \rangle \iff \llbracket c \rrbracket \sigma = \langle \mathbf{abort}, \sigma' \rangle$ .

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  $F^n(\perp)\sigma \neq \perp$ .

- (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

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

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