

Definiciones

miércoles, 17 de abril de 2024

9:39

Lenguaje imperativo con fallas:

$\langle \text{comm} \rangle ::= \dots \mid \text{fail} \mid \text{catchin } \langle \text{comm} \rangle \text{ with } \langle \text{comm} \rangle$

$$\hat{\Sigma} = \Sigma \cup (\{\text{abort}\} \times \Sigma)$$

$$\square_* : \Sigma \rightarrow \hat{\Sigma}_\perp \quad \hat{\Sigma}_\perp \rightarrow \hat{\Sigma}_\perp$$

$$f_* \sigma = \begin{cases} \sigma \in \Sigma & \rightarrow f \sigma \\ \text{si no} & \rightarrow \sigma \end{cases}$$

$$\square_\dagger : (\Sigma \rightarrow \Sigma) \rightarrow \hat{\Sigma}_\perp \rightarrow \hat{\Sigma}_\perp$$

$$f_\dagger \sigma = \begin{cases} \sigma \in \Sigma & \rightarrow f \sigma \\ \sigma \in (\{\text{abort}\} \times \Sigma) & \rightarrow \langle \text{abort}, f \sigma_1 \rangle \\ \text{si no} & \rightarrow \perp \end{cases}$$

$$\square_+ : \Sigma \rightarrow \hat{\Sigma}_\perp \rightarrow \hat{\Sigma}_\perp$$

$$f_+ \sigma = \begin{cases} \sigma \in (\{\text{abort}\} \times \Sigma) & \rightarrow f \sigma_2 \\ \text{si no} & \rightarrow \sigma \end{cases}$$

La definición es la misma de antes pero cambiando los Σ_\perp por $\hat{\Sigma}_\perp$ y los \perp por $*$ salvo en newvar en donde se lo cambia por \dagger y agregando:

$\text{fail} = \langle \text{abort}, \sigma \rangle$

$\text{catchin } c_0 \text{ with } c_1 \text{ for } \sigma = [c_1][[c_0]]\sigma$

1)

miércoles, 17 de abril de 2024 9:54

(1) Dado el programa $P \equiv$

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

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

$$\{\sigma \in \Sigma : \sigma x + \sigma y \leq 0\}$$

2) 

miércoles, 17 de abril de 2024 9:58

(2) Demostrar o refutar las siguientes equivalencias usando semántica denotacional:

$$(a) c; \text{while true do skip} \equiv \text{while true do skip}.$$

$$(b) c; \text{fail} \equiv \text{fail}.$$

$$(c) \text{newvar } v := e \text{ in } v := v + 1; \text{fail} \equiv$$

$$\text{newvar } w := e \text{ in } w := w + 1; \text{fail}.$$

$$(d) \text{while } b \text{ do fail} \equiv \text{if } b \text{ then fail else skip}.$$

$$(e) x := 0; \text{catch } x := 1 \text{ in while } x < 1 \text{ do fail} \equiv$$

$$x := 0; \text{while } x < 1 \text{ do catch } x := 1 \text{ in fail}$$

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

a)

Es falso si $\llbracket c \rrbracket \in \{abort\} \times \Sigma$

Para lenguaje sin fallas posiblemente si sea cierto

b)

Es falso en el caso en que $\llbracket c \rrbracket = (abort, \sigma') \text{ con } \sigma' \neq \sigma$

c)

$$\text{newvar } v := e \text{ in } v := v + 1; \text{fail} \equiv$$

=

$$rest_{v, \sigma_+} \langle \llbracket v := v + 1; \text{fail} \rrbracket \mid v : e \rangle$$

=

$$rest_{v, \sigma_+} \langle \llbracket v := v + 1 \rrbracket \mid v : e \rangle$$

=

$$rest_{v, \sigma_+} \langle \llbracket \text{fail} \rrbracket \mid v : e \rangle$$

=

$$rest_{v, \sigma_+} \langle \llbracket \text{abort}, [\sigma \mid v : e] + 1 \rrbracket \rangle$$

=

$$\langle \llbracket \text{abort}, rest_{v, \sigma} [\sigma \mid v : e] + 1 \rrbracket \rangle$$

=

$\langle \text{abort}, \sigma \rangle$

Análogamente se puede ver que el otro es igual a esto