Trabajo Práctico 1

Análisis de lenguajes de programación

LCC

Manuel Spreutels
Augusto Rabbia

Septiembre, 2024

1 Soluciones

1.1 Ejercicio 1

Se agregan las siguiente reglas:

$$\underbrace{\begin{aligned} intexp \ &::= \ var -- \\ &\mid var ++ \end{aligned}}_{\text{Sintaxis abstracta}} \qquad \underbrace{\begin{aligned} intexp \ &::= \ var \text{ '}--\text{'} \\ &\mid var \text{ '++'} \end{aligned}}_{\text{Sintaxis concreta}}$$

1.2 Ejercicio 2

Se agregaron en el archivo AST.hs los siguientes constructores:

- VarDec :: Variable $\rightarrow$ Exp Int
- VarInc :: Variable $\rightarrow$ Exp Int

1.3 Ejercicio 3

En el archivo Parser.hs

1.4 Ejercicio 4

Se agregan a la semántica operacional Big-Step para expresiones las siguientes reglas:

$$\frac{x \in dom\ \sigma \qquad \langle x, \sigma\rangle \Downarrow_{exp} \langle n, \sigma\rangle}{\langle x--, \sigma\rangle \Downarrow_{exp} \langle n-1, [\sigma|x : n-1]\rangle} \text{ EVARDEC}$$

$$\frac{x \in dom\ \sigma \qquad \langle x, \sigma\rangle \Downarrow_{exp} \langle n, \sigma\rangle}{\langle x++, \sigma\rangle \Downarrow_{exp} \langle n+1, [\sigma|x : n+1]\rangle} \text{ EVARINC}$$

1.5 Ejercicio 5

Suponemos $\alpha \rightsquigarrow \beta$ y $\alpha \rightsquigarrow \gamma$. Hacemos inducción sobre $\alpha \rightsquigarrow \beta$. Supongamos que la última regla utilizada fue:

- **ASS**.
  Luego, sabemos:

  a) $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$

b) $\alpha = \langle v = e, \sigma \rangle$

c) $\beta = \langle \textbf{skip}, [\sigma' | v : n] \rangle$

Analizamos $\alpha \rightsquigarrow \gamma$:

La regla aplicada debe ser **ASS** por la forma de α. Es decir, $\alpha \rightsquigarrow \gamma$ tiene la forma

$$\frac{\langle e, \sigma \rangle \Downarrow_{exp} \langle n', \sigma'' \rangle}{\langle v = e, \sigma \rangle \rightsquigarrow \langle \textbf{skip}, [\sigma'' | v : n] \rangle} \text{ASS}$$

Luego, como $\Downarrow$ es determinista, se tiene $\langle n, \sigma' \rangle = \langle n', \sigma'' \rangle$ y por lo tanto,
$\gamma = \langle \textbf{skip}, [\sigma'' | v : n'] \rangle = \langle \textbf{skip}, [\sigma' | v : n] \rangle = \beta$

- **SEQ1**.
  Sabemos:

  a) $\alpha = \langle \textbf{skip}; c_1, \sigma \rangle$

  b) $\beta = \langle c_1, \sigma \rangle$

  Analizamos $\alpha \rightsquigarrow \gamma$: la regla aplicada debe ser **SEQ1** puesto que **skip** es el primer comando de la secuencia en α. Es decir, $\alpha \rightsquigarrow \gamma$ tiene la forma

  $$\frac{}{\langle \textbf{skip}; c_1, \sigma \rangle \rightsquigarrow \langle c_1, \sigma \rangle} \text{SEQ1}$$

  Luego, $\gamma = \langle c_1, \sigma \rangle = \beta$.

- **IF1**.
  Se tiene:

  a) $\langle b, \sigma \rangle \Downarrow \langle \textbf{true}, \sigma' \rangle$

  b) $\alpha = \langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma \rangle$

  c) $\beta = \langle c_0, \sigma' \rangle$

  Analizamos $\alpha \rightsquigarrow \gamma$: observemos que por la forma de α, solo pueden aplicarse las reglas **IF1** e **IF2** para derivar $\alpha \rightsquigarrow \gamma$. Supongamos que se aplicó **IF2**:

  $$\frac{\langle b, \sigma \rangle \Downarrow \langle \textbf{false}, \sigma'' \rangle}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma \rangle \rightsquigarrow \langle c_1, \sigma'' \rangle} \text{IF2}$$

  Se deduce que $\langle b, \sigma \rangle \Downarrow \langle \textbf{false}, \sigma'' \rangle$, pero $\Downarrow$ es determinista $\overset{\textbf{a)}}{\Longrightarrow}$ absurdo. Luego, debió aplicarse **IF1** para $\alpha \rightsquigarrow \gamma$:

$$\frac{\langle b, \sigma\rangle \Downarrow \langle \textbf{true}, \sigma''\rangle}{\langle \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, \sigma\rangle \rightsquigarrow \langle c_0, \sigma''\rangle} \text{ IF1}$$

Pero como $\Downarrow$ es determinista, asi que $\sigma'' = \sigma'$, y por tanto $\gamma = \langle c_0, \sigma''\rangle = \langle c_0, \sigma'\rangle = \beta$

- **IF2**.
  Análogo al caso **IF1**

- **REPEAT**.
  Se tiene:

  a) $\alpha = \langle \textbf{repeat } c \textbf{ until } b, \sigma\rangle$

  b) $\beta = \langle c;\ \textbf{if } b \textbf{ then skip else repeat } c \textbf{ until } b, \sigma\rangle$

  Analizamos $\alpha \rightsquigarrow \gamma$: la regla aplicada debe ser **REPEAT** por la forma de α. Es decir, $\alpha \rightsquigarrow \gamma$ tiene la forma

$$\frac{}{\langle \textbf{repeat } c \textbf{ until } b, \sigma\rangle \rightsquigarrow \langle c;\ \textbf{if } b \textbf{ then skip else repeat } c \textbf{ until } b, \sigma\rangle} \text{ REPEAT}$$

  Luego, $\gamma = \langle c;\ \textbf{if } b \textbf{ then skip else repeat } c \textbf{ until } b, \sigma\rangle = \beta$.

- **SEQ2**.
  Sabemos entonces:

  a) $\langle c_0, \sigma\rangle \rightsquigarrow \langle c_0', \sigma'\rangle$

  b) $\alpha = \langle c_0; c_1, \sigma\rangle$

  c) $\beta = \langle c_0'; c_1, \sigma'\rangle$

  HI: Suponemos que para toda subderivación de $\alpha \rightsquigarrow \beta$, de la forma $\alpha' \rightsquigarrow \beta'$, si $\alpha' \rightsquigarrow \beta'$ y $\alpha' \rightsquigarrow \beta''$, entonces $\beta' = \beta''$.
  Por a), sabemos que $c_0 \neq$ **skip**, pues dadas las reglas de evaluación small-step que se tienen, $\nexists c_0'/\langle \textbf{skip}, \sigma\rangle \rightsquigarrow \langle c_0', \sigma'\rangle$. Esto, junto con la forma de α, nos lleva a concluir que sólo **SEQ2** pudo aplicarse para la derivación de $\alpha \rightsquigarrow \gamma$. Conocemos entonces la forma de esta derivación:

$$\frac{\langle c_0, \sigma\rangle \rightsquigarrow \langle c_0'', \sigma''\rangle}{\langle c_0; c_1, \sigma\rangle \rightsquigarrow \langle c_0''; c_1, \sigma''\rangle} \text{ SEQ2}$$

  Luego, $\gamma = \langle c_0''; c_1, \sigma''\rangle \overset{\textbf{(HI)}}{=} \langle c_0'; c_1, \sigma'\rangle = \beta$.

1.6 Ejercicio 6

Lo probamos desarrollando sus árboles de derivación. Para cualquier estado σ tal que $x \in dom\ \sigma$, para el primer programa se tiene :

- **Paso 1**.

$$\dfrac{\text{PLUS}\ \dfrac{\dfrac{x \in dom\ \sigma}{\langle x, \sigma\rangle \Downarrow_{exp} \langle \sigma\ x, \sigma\rangle}\ \text{Var} \qquad \dfrac{}{\langle 1, \sigma\rangle \Downarrow_{exp} \langle 1, \sigma\rangle}\ \text{NVAL}}{\langle x+1, \sigma\rangle \Downarrow_{exp} \langle \sigma\ x+1, \sigma\rangle}}{\langle x = x+1; y = x, \sigma\rangle \rightsquigarrow \langle \mathbf{skip}; y = x, [\sigma | x : \sigma\ x+1]\rangle}\ \text{ASS}$$

- **Paso 2**.

$$\dfrac{}{\langle \mathbf{skip}; y = x, [\sigma | x : \sigma x+1]\rangle \rightsquigarrow \langle y = x, [\sigma | x : \sigma\ x+1]\rangle}\ \text{SEQ1}$$

- **Paso 3**.

$$\dfrac{\dfrac{x \in dom\ [\sigma | x : \sigma\ x+1]}{\langle x, [\sigma | x : \sigma\ x+1]\rangle \Downarrow_{exp} \langle \sigma\ x+1, [\sigma | x : \sigma\ x+1]\rangle}\ \text{Var}}{\langle y = x, [\sigma | x : \sigma\ x+1]\rangle \rightsquigarrow \langle \mathbf{skip}, [\sigma | x : \sigma\ x+1, y : \sigma\ x+1]\rangle}\ \text{ASS}$$

Ahora, hacemos lo mismo para el segundo programa:

- **Paso 1**.

$$\dfrac{\dfrac{x \in dom\ \sigma \qquad \dfrac{x \in dom\ \sigma}{\langle x, \sigma\rangle \Downarrow_{exp} \langle \sigma\ x, \sigma\rangle}\ \text{Var}}{\langle x++, \sigma\rangle \Downarrow_{exp} \langle \sigma\ x+1, [\sigma | x : \sigma\ x+1]\rangle}\ \text{EVARINC}}{\langle y = x++, \sigma\rangle \rightsquigarrow \langle \mathbf{skip}, [\sigma | x : \sigma\ x+1, y : \sigma\ x+1]\rangle}\ \text{ASS}$$

Por lo tanto, vemos que $\langle x = x+1; y = x, \sigma\rangle \rightsquigarrow^* \langle \mathbf{skip}, [\sigma | x : \sigma\ x+1, y : \sigma\ x+1]\rangle$ y $\langle y = x++, \sigma\rangle \rightsquigarrow^* \langle \mathbf{skip}, [\sigma | x : \sigma\ x+1, y : \sigma\ x+1]\rangle$ para cualquier σ tal que $x \in dom\ \sigma$, y por lo tanto ambos programas son equivalentes.

1.7 Ejercicio 7

En el archivo Eval1.hs

1.8 Ejercicio 8

En el archivo Eval2.hs

1.9 Ejercicio 9

En el archivo Eval3.hs