Semántica operacional básica. Ejercicios

Ejercicio 2.8

Enunciado

Extender el lenguaje **WHILE** con la instrucción `for` :

```
for x := a1 to a2 do S
```

Definir la relación $\to$ para esta construcción.

Resolución

Veamos en primer lugar la definición cuando se entra en el bucle:

$$\left[\text{for}_{\text{ns}}^{\text{tt}}\right] := \frac{\langle S, s\,[x \mapsto \mathcal{A}[\![a_1]\!]s]\rangle \to s', \langle \mathtt{for}\ x := x+1\ \mathtt{to}\ a_2\ \mathtt{do}\ S, s'\rangle \to s''}{\langle \mathtt{for}\ x := a_1\ \mathtt{to}\ a_2\ \mathtt{do}\ S, s\rangle \to s''},$$
$$\text{si } \mathcal{B}[\![a_1 \le a_2]\!]s = \mathbf{tt}.$$

Y ahora el caso en que no se cumpla la condición:

$$\left[\text{for}_{\text{ns}}^{\text{ff}}\right] := \langle \mathtt{for}\ x := a_1\ \mathtt{to}\ a_2\ \mathtt{do}\ S, s\rangle \to s,\ \text{si } \mathcal{B}[\![a_1 \le a_2]\!]s = \mathbf{ff}$$

El caso de interés es cuando se cumple la condición. Los aspectos más destacables son los siguientes:

- En las premisas tenemos, en primer lugar, que el estado s' se obtiene de ejecutar S sobre el estado original s después de aplicar la asignación a x con a_1.
- La definición no puede ser composicional puesto que tenemos que capturar la «iteratividad» de la instrucción. Por eso, en las premisas tenemos un `for` , pero con la peculiaridad de que le asignamos una unidad más de su valor en el estado s'. Esto es una decisión de diseño que podría hacerse de distinta manera, pero así aseguramos que el bucle acabe en algún momento.

Ejercicio 2.10

Enunciado

Demostrar que la construcción

```
repeat S until b
```

Es semánticamente equivalente a `S; while ¬b do S`.

Resolución

En primer lugar, la definición para la semántica de paso largo del `repeat` es:

$$\left[\text{repeat}^{\text{tt}}_{\text{ns}}\right] := \frac{\langle S, s\rangle \to s'}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'},\ \text{si}\ \mathcal{B}[\![b]\!]s' = \mathbf{tt}$$
$$\left[\text{repeat}^{\text{ff}}_{\text{ns}}\right] := \frac{\langle S, s\rangle \to s', \langle \texttt{repeat}\ S\ \texttt{until}\ b, s'\rangle \to s''}{\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s''},\ \text{si}\ \mathcal{B}[\![b]\!]s' = \mathbf{ff}$$

Ahora veremos la equivalencia usando inducción sobre el árbol de derivación. Sea $s \in \mathbf{State}$.

- En primer lugar, supongamos que $\langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \to s'$. Queremos ver ahora que $\langle S\ ;\ \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s'$, o lo que es lo mismo (por definición de composición), $\langle S, s\rangle \to s_0$ y $\langle \texttt{while}\ \neg b\ \texttt{do}\ S, s_0\rangle \to s'$. La primera premisa la tenemos directamente por definición de `repeat`. Ahora distinguimos casos:
  - Si $\mathcal{B}[\![b]\!]s_0 = \mathbf{tt}$ quiere decir que $s_0 \equiv s'$ y que, por definición del `while` y $\mathcal{B}[\![\neg b]\!]s_0 = \mathbf{ff}$, $\langle \texttt{while}\ \neg b\ \texttt{do}\ S, s_0\rangle \to s_0 \equiv s'$. Con esto, para este caso se cumple la implicación.
  - Si ahora $\mathcal{B}[\![b]\!]s_0 = \mathbf{ff}$, tenemos que, para demostrar $\langle \texttt{while}\ \neg b\ \texttt{do}\ S, s_0\rangle \to s'$, necesitamos ver que $\langle S, s_0\rangle \to s_1$ y que $\langle \texttt{while}\ \ b\ \texttt{do}\ S, s_1\rangle \to s'$ o lo que es lo mismo, $\langle S\ ;\ \texttt{while}\ \neg b\ \texttt{do}\ S, s_0\rangle \to s'$. Sin embargo, por la definición de `repeat` tenemos $\langle \texttt{repeat}\ S\ \texttt{until}\ b, s_0\rangle \to s'$ con lo que aplicando la hipótesis de inducción tenemos el resultado.
- Se ahora suponemos la otra hipótesis, $\langle S\ ;\ \texttt{while}\ \neg b\ \texttt{do}\ S, s\rangle \to s'$ simplemente tendremos que invertir los pasos dados en el anterior apartado para sacar el resultado final.

Ejercicio 2.11

Enunciado

Definir la semántica de la expresiones aritméticas $\mathcal{A}$ de forma similar a la semántica operacional de paso largo.

Demostrar que es equivalente a la anteriormente definida.

Resolución

Definición:

- Constantes:

$$\langle n, s\rangle \rightarrow_{\text{Aexp}} \mathcal{N}[\![n]\!]$$

- Variables:

$$\langle x, s\rangle \rightarrow_{\text{Aexp}} s\ x$$

- Sumas:

$$\frac{\langle a_1, s\rangle \rightarrow_{\text{Aexp}} z_1, \langle a_2, s\rangle \rightarrow_{\text{Aexp}} z_2}{\langle a_1 + a_2, s\rangle \rightarrow_{\text{Aexp}} z}, \ \ \text{con } z_1 + z_2 = z$$

- Restas:

$$\frac{\langle a_1, s\rangle \rightarrow_{\text{Aexp}} z_1, \langle a_2, s\rangle \rightarrow_{\text{Aexp}} z_2}{\langle a_1 - a_2, s\rangle \rightarrow_{\text{Aexp}} z}, \ \ \text{con } z_1 - z_2 = z$$

- Productos:

$$\frac{\langle a_1, s\rangle \rightarrow_{\text{Aexp}} z_1, \langle a_2, s\rangle \rightarrow_{\text{Aexp}} z_2}{\langle a_1 * a_2, s\rangle \rightarrow_{\text{Aexp}} z}, \ \ \text{con } z_1 * z_2 = z$$

Para la demostración usaremos inducción estructural sobre $\mathbf{Aexp}$. El caso compuesto (las operaciones) lo haremos genérico ya que son iguales. Empezamos con los casos base. Por simple inspección y definición de $\mathcal{A}$, son equivalentes. Veamos el caso inductivo. Utilizaremos $\times$ como cualquiera de los operadores. Por hipótesis de inducción, suponemos que $z_1 \equiv \mathcal{A}[\![a_1]\!]s$ y $z_2 \equiv \mathcal{A}[\![a_2]\!]s$. Con esto, $z = \mathcal{A}[\![a_1]\!]s \times \mathcal{A}[\![a_2]\!]s$, pero esto último es, por definición de $\mathcal{A}$, igual a $\mathcal{A}[\![a_1 \times a_2]\!]s$ con lo que tenemos el resultado.

Ejercicio 2.17

Enunciado

Dar una semántica de paso corto para la instrucción `repeat` .

Resolución

Tenemos la siguiente definición:

$$[\text{repeat}_{\text{sos}}] := \langle \texttt{repeat}\ S\ \texttt{until}\ b, s\rangle \Rightarrow \langle S\ ;\ \texttt{if}\ b\ \texttt{then}\ \texttt{skip}\ \texttt{else}\ (\texttt{repeat}\ S\ \texttt{until}\ b)\, , s\rangle$$

Ejercicio 2.18

Enunciado

Dar una semántica de paso corto para la construcción `for` .

Resolución

Tenemos la siguiente definición:

$$[\text{for}_{\text{sos}}] := \langle \texttt{for } x := a_1 \texttt{ to } a_2 \texttt{ do } S, s\rangle \Rightarrow$$
$$\langle \texttt{if } a_1 \leq a_2 \texttt{ then } (S \texttt{ ; for } x := x + 1 \texttt{ to } a_2 \texttt{ do } S) \texttt{ else skip}, s\rangle$$

Es decir, si el valor de x tras la asignación del inicio del `for` es inferior al valor de la expresión límite, ejecutamos una iteración. Para la siguiente iteración comenzamos sumando uno a x.

Ejercicio 2.23

Enunciado

Mostrar la equivalencia semántica (paso corto) entre las siguientes parejas de sentencias:

1. $S \texttt{ ; skip}$ y S.
2. $\texttt{while } b \texttt{ do } S$ y $\texttt{if } b \texttt{ then } (S \texttt{ ; while } b \texttt{ do } S) \texttt{ else skip}$.
3. $S_1 \texttt{ ; } (S_2 \texttt{ ; } S_3)$ y $(S_1 \texttt{ ; } S_2) \texttt{ ; } S_3$

Resolución

Tenemos que demostrar para cada caso que:

$$\langle S_1, s\rangle \Rightarrow^* \gamma \Leftrightarrow \langle S_2, s\rangle \Rightarrow^* \gamma,$$

o que si uno tiene una secuencia infinita de derivación, el otro también.

1. Sea $s \in \mathbf{State}$. Razonemos por inducción sobre la longitud de la secuencia.

   - **Caso base**: El caso $k = 0$ no se puede dar directamente, pero el caso $k = 1$ solo podrá salir de $\left[\text{comp}^2_{\text{sos}}\right]$, es decir, $\langle S \texttt{ ; skip}, s\rangle \Rightarrow \langle \texttt{skip}, s'\rangle \Rightarrow s'$. Como se da esto, también se darán las premisas de este derivación, es decir, $\langle S, s\rangle \Rightarrow s'$ con lo que tenemos el resultado buscado.
   - **Caso inductivo**: Sea $k > 1$, es decir, $\langle S \texttt{ ; skip}, s\rangle \Rightarrow^k \gamma$. Como $k > 1$, solo podemos aplicar $\left[\text{comp}^1_{\text{sos}}\right]$, es decir, $\langle S \texttt{ ; skip}, s\rangle \Rightarrow \langle S' \texttt{ ; skip}, s'\rangle$. Con esto ahora tendremos que $\langle S' \texttt{ ; skip}, s'\rangle \Rightarrow^{k-1} \gamma$, con lo que podemos aplicar la hipótesis de inducción y ver que $\langle S', s'\rangle \Rightarrow^{k-1} \gamma$. Sin embargo, al aplicar la anterior derivación, también serán ciertas las hipótesis, es decir, $\boxed{\langle S, s\rangle \Rightarrow \langle S', s'\rangle} \Rightarrow^{k-1} \gamma$ con lo que tendremos el resultado.
2. Trivialmente lo vemos por la definición del `while` .

3. Sea $s \in \mathbf{State}$. Razonaremos por inducción sobre la longitud de la cadena.

- **Caso base**. Si $k = 1$, solo podemos aplicar la derivación $\left[\text{comp}^2_{\text{sos}}\right]$ con lo que tenemos que $\langle S_1 \ ; \ (S_2 \ ; \ S_3)\,, s\rangle \Rightarrow \langle S_2 \ ; \ S_3, s'\rangle$ y que $\langle S, s\rangle \Rightarrow s'$. Pero, a su vez, si se cumple esto, tenemos que $\langle S_1 \ ; \ S_2, s\rangle \Rightarrow \langle S_2, s'\rangle$ con lo que $\langle (S_1 \ ; \ S_2) \ ; \ S_3, s\rangle \Rightarrow \langle S_2 \ ; \ S_3, s'\rangle$
- **Caso inductivo**. Si $k > 1$, tenemos que aplicar $\left[\text{comp}^1_{\text{sos}}\right]$ con lo que tenemos $\langle S_1 \ ; \ (S_2 \ ; \ S_3)\,, s\rangle \Rightarrow \langle S_1' \ ; \ (S_2 \ ; \ S_3)\,, s'\rangle$ y la premisa $\langle S_1, s\rangle \Rightarrow \langle S_1', s'\rangle$. Como antes, con esta premisa aplicada a la otra sentencia, tenemos que $\langle (S_1 \ ; \ S_2) \ ; \ S_3, s\rangle \Rightarrow \langle (S_1' \ ; \ S_2) \ ; \ S_3, s'\rangle$. Sin embargo, como la longitud de la secuencia para llegar a γ desde $\langle S_1' \ ; \ (S_2 \ ; \ S_3)\,, s'\rangle$ es $k-1$, podemos aplicar la hipótesis de inducción, obtenemos que $S_1' \ ; \ (S_2 \ ; \ S_3)$ y $(S_1' \ ; \ S_2) \ ; \ S_3$ son equivalentes y tenemos el resultado que buscábamos.